ENGINEERING

ADA086339

LEVEL II

DDC FILE COPY

USCIPI Report 960

12

UNIVERSITY OF SOUTHERN CALIFORNIA

IMAGE UNDERSTANDING RESEARCH

Semiannual Technical Report

Ramakant Nevatia
Alexander A. Sawchuk
Principal Investigators

Covering Research Activity During the Period
1 October 1979 through 31 March 1980

31 March 1980

Image Processing Institute
University of Southern California
University Park
Los Angeles, California 90007

Sponsored by

Contract No. F-33615-76-C-1203
DARPA Order No. 3119

DTIC
ELECTE
JUN 30 1980
S D
C

This document has been approved for public release and sale; its distribution is unlimited.

IPI

IMAGE PROCESSING INSTITUTE

80 6 27 013

USCIPI Report 960

IMAGE UNDERSTANDING RESEARCH

Semiannual Technical Report

Covering Research Activity During the Period
1 October 1979 through 31 March 1980

Ramakant Nevatia
Alexander A. Sawchuk
Principal Investigators
(213) 741-5506

Image Processing Institute
University of Southern California
University Park
Los Angeles, California 90007

31 March 1980

This document has been approved for public release and sale; its distribution is unlimited.

This research was supported by the Defense Advanced Research Projects Agency and was monitored by the Wright Patterson Air Force Base under Contract F-33615-76-C-1203, DARPA Order No. 3119.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: USCIPI Report 960	2. GOVT ACCESSION NO.: AD-A086339
3. RECIPIENT'S CATALOG NUMBER: Technical
4. TITLE (and Subtitle): IMAGE UNDERSTANDING RESEARCH -	5. TYPE OF REPORT & PERIOD COVERED: Semiannual Report, 1 Oct 79 — 31 March 80
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (Principal Investigators): Ramakant Nevatia, Alexander A. Sawchuk	8. CONTRACT OR GRANT NUMBER(s): F 33615-76-C-1203, DARPA Order-3119
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Image Processing Institute, University of Southern California, University Park, Los Angeles, Ca. 90007	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: DARPA Order No. 3119
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency, 1400 Wilson Boulevard, Arlington, Virginia 22209	12. REPORT DATE: March 1980; 13. NUMBER OF PAGES: 185
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Wright Patterson Air Force Base, U.S. Air Force, Air Force Avionics Laboratory, Wright Patterson AFB, Ohio 45433	15. SECURITY CLASS. (of this report): UNCLASSIFIED; 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for release: distribution unlimited

12 184

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

14 USCIPI-960

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Key Words: Digital Image Processing, Image Restoration, Scene Analysis, Image Understanding, Edge Detection, Image Segmentation, CCD Arrays, CCD Processors, VLSI Processors.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This technical report summarizes the image understanding and smart sensor and VLSI hardware research activities performed by the USC Image Processing Institute and the Hughes Research Laboratories during the period of 1 October 1979 through 31 March 1980 under contract number F-33615-76-C-1203 with the Defense Advanced Research Projects Agency, Information Processing

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

391141

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Techniques Office, and monitored by the Wright-Patterson Air Force Base, Dayton, Ohio.

The research program has, as its primary purpose, the development of techniques and systems for understanding images. Methodologies range from low level image processing principles, smart sensor CCD LSI and VLSI circuit design, up to higher level symbolic representations and relational structure manipulations.

ABSTRACT

This technical report summarizes the image understanding, smart sensor and VLSI hardware research activities performed by the USC Image Processing Institute and the Hughes Research Laboratories during the period of 1 October 1979 through 31 March 1980 under contract number F-33615-76-C-1203 with the Defense Advanced Research Projects Agency, Information Processing Techniques Office, and monitored by the Wright-Patterson Air Force Base, Dayton, Ohio.

The research program has, as its primary purpose, the development of techniques and systems for understanding images. Methodologies range from low level image processing principles, smart sensor CCD LSI and VLSI circuit design, up to higher level symbolic representations and relational structure manipulations.

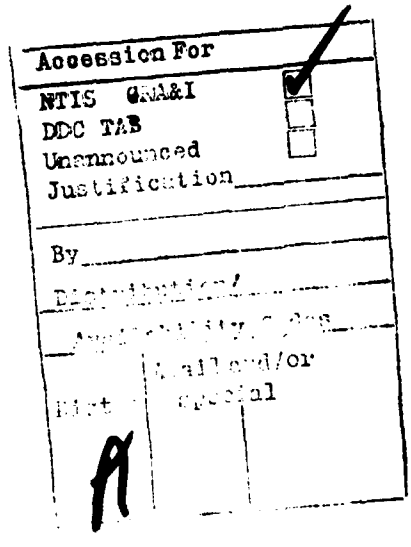


TABLE OF CONTENTS

1. RESEARCH OVERVIEW

We have continued work at various levels of our IU system and started to evaluate techniques for applications to DMA supplied images. This report constitutes the final report on contract F-33615-76-C-1203. The last six months work can be described under the following headings:

Image Matching

Matching of an image to a symbolic map, or a symbolic description of another image, is central for the tasks of map updating and change detection. In the past we have described results using aerial images of areas such as San Francisco, Stockton, San Diego, etc. [1]. These previous techniques used a simple matching scheme, with each element in one description being matched to the best corresponding element in the other description, and not allowing for any revision based on the matching of other neighboring elements. We have now incorporated a relaxation matching algorithm. This algorithm is different from those used by Rosenfeld and associates [2], in the use of a well defined optimization criterion.

Texture Analysis

We have continued development of our structural texture analysis techniques. The analysis uses micro-edges detected in a texture and derives repetition pattern characteristics of these edges. Our previous presentations have described techniques for determining the width of texture primitives and a repetition period, if any. Our new techniques are also able to extract the length of the primitives and thus describe the shapes of the primitives. The usefulness of these descriptions for recognition of natural textures is currently being tested.

Texture Synthesis

We have several ongoing projects in synthesis of natural textures from a stochastic model using few parameters. The techniques include auto-regressive modeling with conditional expectations and algebraic reconstruction techniques. Models for texture synthesis are also useful for texture analysis, as they validate the sufficiency of the models used for analysis.

Segmentation

We are trying to use the texture analysis techniques to aid in scene segmentation. Texture features can be used as intensity features for segmentation. However, difficulties arise because texture features must be measured over a window assumed to contain a single texture only. Also, texture features have many components and should be treated as a vector. Faugeras and Lee give some preliminary results in Section 2.4.

Another segmentation project is to develop techniques for segmenting images that have unimodal intensity (or color) histograms. This happens typically when the image consists mostly of a large background region, with small but significant other regions. Faugeras and Bhanu have developed a gradient relaxation technique to modify the histogram to bring out the peaks corresponding to the small regions (Section 2.9). Currently, this technique is applicable if only two types of regions are present in the image.

Range Data Processing

Our previous linear feature extraction techniques, developed for intensity image processing [3], have also proved useful for range data processing. However, for range data we are able to obtain complete or near complete object boundaries by subsequent processing that exploits the richer information contained in such data. The DARPA contract has

only supplied minimal computing funds for this work. However, it is included here because of its potential application to aerial image processing.

Hardware Implementation

In continuing work with Hughes Research Laboratories, Malibu, California, we are investigating the use of VLSI technology for hardware implementation of IU algorithms. We have chosen to investigate the following algorithms initially:

i) Nevatia-Babu Line Finder [3]
ii) Ohlander Region Segmentor [4]
iii) Laws Texture Analysis System [5]

The choice of the above three algorithms was based on their computation intensive nature, their use for a broad range of problems and experience with a large number of images for the first two. Also these algorithms are largely local and hence easier to implement in VLSI hardware, where reducing interconnections is important. Further, the three algorithms have common kernels, such as convolution, but also require different subsequent processing. A study of there should provide valuable feedback on the feasibility of hardware implementation for a large class of algorithms.

At this time, no decision on algorithms for actual implementation has been made and opinions of the IU community are invited on the suitability of the proposed algorithms as well as suggestions for other algorithms.

References

1. R. Nevatia and K. Price, "Locating Structures in Aerial Images," Proceedings of ARPA Image Understanding Workshop, Palo Alto, Ca., October 1977.

2. A. Rosenfeld, R.A. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. on Systems, Man & Cybernetics, SMC-6, No. 6, pp. 420-453, June 1976.

3. R. Nevatia and K.R. Babu, "Linear Feature Extraction," Proceedings of the ARPA Image Understanding Workshop, Pittsburgh, Pa., Nov. 1978, pp. 73-78.

4. R. Ohlander, K. Price and R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," Computer Graphics and Image Processing, Vol. 8, 1978, pp. 313-333.

5. K. Laws, "Textured Image Segmentation," USCIPI Report 940, January 1980.

2. IMAGE UNDERSTANDING PROJECTS

2.1 Semantic Description of Aerial Images Using Stochastic Labeling

O.D. Faugeras and K.E. Price

Introduction

The purpose of computer scene analysis is to automatically produce a description of the content of an image similar to one obtained from a skilled human observer. In order to achieve such a goal, a symbolic description of the raw image data must be constructed. This requires the application of many of the now well developed techniques of Image Processing (image bandwidth compression, image restoration and image enhancement), extraction of features such as texture and edges, segmentation of the image into homogeneous regions with respect to one or several properties, and measuring features that characterize these segments (color, brightness, texture, size, and shape) and also relations between these segments (brighter than, larger than, above, below, etc.). The output of this complex sequence of processes is something that does not resemble the original input array of pixels but is much more suitable for high level processing: a symbolic description which is represented as a labelled graph or semantic network. To proceed any further, we must also assume that we have access to another body of knowledge containing a priori information about the expected content of images of a given area. We will not make any assumptions about how this world model has been obtained (manual input or intelligent learning) and will only assume that its representation is the same as the image, i.e., that it is also a semantic network. The process of obtaining a semantic description of a given image can then be viewed as finding the solution of a graph matching problem: either match the image onto the model or match the model onto the image. Thus, this is equivalent to

the general graph/subgraph isomorphism problem which is well known to be NP-complete. Practical and useful solutions can nonetheless be found as we will show in this paper.

Matching techniques must be described in terms of performance on a well defined problem. The particular task under study is the analysis of an image of a scene using an approximate specification of the scene which would apply for many different images of the scene (i.e. a model). Both the image and scene are represented by semantic networks, the image description is automatically generated and thus reflects any errors in the segmentation process. The model is specified by the user and contains only the important objects and relations.

A similar problem was attacked by Rubin [1] with a search procedure and a more detailed model. His work has been combined with a relaxation procedure by Smith [2]. The general form of the solution is similar to ours but the scene model is significantly more detailed so that exact comparisons cannot be made.

We will first present the image and model descriptions, which are the input to the matching procedure, then the basic matching technique which we use to compare two objects. Next, the global matching process (labeling) will be described, finally we will discuss the results of applying this procedure.

Description and Basic Matching Techniques

This section will first present a description of the symbolic representation used for the scene and image. Then the method by computing initial likelihoods of particular matches is described along with the method to compute compatibilities of pairs of matches.

Image And Model Descriptions

Our matching system uses a feature based, symbolic, description of an idealized scene (the model) and of the image of a portion of this scene [3]. The image description is derived automatically from an image and the model is developed by the user through an interactive procedure.

The basic objects used for the image description are the segments of the image generated both by a general region based image segmentation procedure [4] and by a linear feature extraction procedure [5]. The regions are derived by locating connected areas which are uniform with respect to some feature in the input image (color parameters, texture values etc.). Linear features are defined as long narrow objects which differ from the background on both sides, and are described by as a sequence of straight line segments with some small width. Typically, the images which we use have a total of 100-200 individual segments of both types. The symbolic description is completed by computing various features of the segments and relations between them.

The features used for the symbolic description are those which can be reliably computed from the available data (the input image and the region or line descriptions). They include properties such as average color and texture (currently only simple texture measures), size, position, two-dimensional orientation, and simple shape measures. Also included are various relations between image segments such as adjacency, nearby, and relative positions (above, below, etc.). With all these relations a segment may easily be related to as many as 100 other segments. This description is not intended to be used for reconstruction of the original image, it is meant to capture the important, observable, information contained in the image.

The model description is identical to that used for the image - feature based descriptions of basic region-like and line-like

elements including relations between them. Additionally, the basic elements in the model are grouped into more complex objects, associated with generic descriptions, and referred to by actual names. The feature values in the model will not correspond exactly to image values, but are approximations of the likely values, so that one model of a scene can be used with many similar images of the same scene. The relations between elements which are included in the model are the "important" ones, that is, if a relation appears in the model description then it is expected to occur in the image description, but, if no relation occurs in the model then nothing may be said about its appearance in the image (negative relations could be used, such as must not be adjacent, etc.). Similarly, only the important objects are described in the model, thus it is not a complete description of the entire scene. Generally, the model description is smaller than the image description containing 20-30 basic elements.

In summary, the input for the matching procedures is the symbolic descriptions of both the input image and a model of the scene. The model description determines the outcome of the matching operations. The image description is automatically derived and may contain errors - especially where simple objects are broken into several pieces. The model description is incomplete, as it contains only important objects, thus most segments in the image (and objects in the actual scene) will not be described by objects in the model.

Basic Matching Technique

The global matching procedure is a stochastic labeling procedure (also called relaxation procedure) which will be explained in detail in the next section. The relaxation technique requires a basic procedure to compare how well a model element agrees with an image segment for its operation. In our previous work in matching pairs of images [2] and in analysis of images using models of the scene [6] we have developed a comparison procedure which can rate the correspondence of an object in one image (or model) with an object in

another image (or model). The basic procedure combines differences in all available features and relations to produce a single rating of the quality of the match. The past experiments indicated that this procedure produces reliable, and generally accurate measures for the differences between two objects (i.e. the model based matching performed accurately on a variety of scenes). The problems with the past matching system arose in the use of these results by the global matching procedure, particularly in requirements for ordering the selection of elements to match and the handling of objects which break into several pieces. (This is discussed in more detail later).

Briefly, the basic matching procedure combines differences in all feature values which are weighted to account for the difference ranges of values (small size differences (1000 pixels) are not as important for large regions but small changes in orientation (0.5 radian) are very significant). Additionally, differences in the number of relations in the model and the number of the same relations between the corresponding image segments are used. All of these components are given a strength (high, medium, low) to control their impact on the final match result (i.e. features known to be marginally useful are given a low strength, and those considered very important are given a high strenth). In the earlier implementation the absolute value of the rating ranged from 0 upward to 10000 or more. These values are converted to the range [0.0, 1.0] by using the reciprocal of the value plus 1.

The stochastic labeling operation uses the matching procedure for two distinct purposes. First, it is necessary to determine the initial likelihood of a particular assignment (i.e. a rating without consideration of neighbors, or in other words use only the unary relations). Second, the compatibility of particular assignments for two objects must be computed (i.e. the rating using the relations between two objects).

The computation of initial likelihoods cannot use relations between objects since their use depends on assignments of model elements to image segments. Therefore, the initial match is limited to feature values (color, size, shape, etc.). A model element is compared with all the image segments using our basic matching procedure. The best matching segments are kept for further analysis, currently up to thirty are used or up to the point where the worst is 1/10 of the best (whichever given the smaller set). Then the match results are scaled so that they sum to 1, to be treated as probabilities in the stochastic labeling procedure.

Clearly, if feature values, alone, are sufficient to locate correct matches, then the process could stop at this point. But, even though features are sufficient for some well defined objects, they do not locate most correspondences by themselves.

After an initial application of the relaxation procedure several assignments may be made. At this point, the computation of initial likelihoods for an object can, and does, use the relations with assigned elements. This means that initial likelihoods are always computed with all available information, initially only feature values then an increasing number of relations. Therefore successive steps which are using more information can more reliably match the less well defined objects.

The compatibility measure computes the effect of making an assignment for one element on the assignment for another element. The interaction between objects and their assignments is through the relations between them, so that the compatibility measure is based on these relations. Here again, the same basic matching procedure is used to compute how well the relations in the model match the relations in the image. The procedure has been modified so that only the relations between the two model elements and between the potential corresponding image elements are considered.

In some implementations of a stochastic labeling procedure, all the possible compatibility measures are computed once at the beginning, but because of the total number of possibilities which would be required, this can not be done. Since only a small fraction of the total number are ever required, they are computed as needed. Clearly, in one experiment, the same value will be computed several times, but the cost is small.

The compatibility measure is computed using the most likely assignments for the second object. The individual matches are weighted by the likelihood of a particular assignment so that more likely assignments contribute more to the result. The number of likely assignments to be considered is determined by an input parameter. The greater the number of assignments, the greater time the procedure will take. Experiments have indicated that using more than one or two assignments does not contribute much to the final results (see the final section). If an object has a firm assignment (i.e. some segment has been selected as corresponding to a model object) then this assignment is included in the compatibility computation in addition to the number of assignments specified by the parameter described above.

Relaxation Algorithm

This section describes the basic stochastic labeling and the optimization techniques algorithms. The second part of the section presents the variations of the basic procedure which were required for this symbolic matching problem.

General Description

Relaxation labeling attempts to efficiently solve a very general problem in Pattern Recognition and Artificial Intelligence: given a set of units U and a set of names N, assign names to units given actual measured features and a world model. There are two broad

classes of Relaxation techniques. The first one, called discrete Relaxation, handles the case where, for every unit, we can know if a name is possible or impossible. Discrete Relaxation is then an efficient way to solve the search problem of finding a labeling of the units. Continuous Relaxation handles the cases where we know more than just whether names are possible or impossible, namely a measure of their likelihood.

In the first case, the world model consists of a binary relation $R \subset (U \times N) \times (U \times N)$ that determines whether assigning name n_1 to unit u_1 is compatible with assigning name n_2 to unit u_2. In the second case, this compatibility is given by a positive number $c(u_1,n_1,u_2,n_2)$, which is small if the compatibility is weak and large if it is strong. Function c is defined in general only over a subset S of $(U \times N) \times (U \times N)$. To every unit u_i and name n_k we can thus associate the set $V_i(k)$ of related units u_j such that there exists a name n_ℓ for which (u_i,n_k,u_j,n_ℓ) is in S.

In this paper, we are solely concerned with continuous Relaxation, where for every unit u_i, there is a corresponding probability vector $\vec{p}_i = [p_i(1)..p_i(L_i)]^T$ where $p_i(k)$ $(1 \leq k \leq L_i)$ measures the probability that unit u_i has the name n_k. The set of all vectors $\vec{p}_i$ is called a stochastic labeling of the set of units U. As proposed in [7,8] we can also define for each unit u_i a compatibility or prediction vector $\vec{q}_i = [q_i(1)...q_i(L_i)]$ that tells us what $\vec{p}_i$ should be, given the probability of assignments $\vec{p}_j$ at neighboring units u_j and the world model embedded in function c. For simplicity we will rewrite $c(u_i,n_k,u_j,n_\ell)$ as $c(i,k,j,\ell)$ in what follows.

As described in [7,8] we can take:

$$q_i(k) = \frac{Q_i(k)}{\sum_{\ell=1}^{L_i} Q_i(\ell)} \qquad (1)$$

where

$$Q_i(k) = \sum_{u_j \text{ in } V_i(\ell)} \sum_{\ell \text{ in } W_j}^{L_j} c(i,k,j,\ell) p_j(\ell) \qquad (2)$$

and W_j is a subset of the possible names for unit u_j. In the experiments reported in [7,8] we took $W_j = \{1,\ldots,L_j\}$; but because of the large number of possible names in this task and for the sake of efficiency we took

$$W_j = \{\text{set of n most likely names}\}$$

usually with n = 1. That is, for every neighbor of a unit we considered only the contribution of the most likely name in Eq. (2).

In [7,8] we proposed to use local measures of consistency and ambiguity of the form

$$\alpha \| \vec{p}_i - \vec{q}_i \|_2^2 + (1-\alpha) H_i \qquad (3)$$

where $\| \cdot \|_2$ is the usual Euclidean norm, H_i the quadratic entropy

$$H_i = \sum_{\ell=1}^{L_i} p_i(\ell)(1-p_i(\ell)) = 1 - \| \vec{p}_i \|_2^2 \qquad (4)$$

and α a weighting factor adjusting the relative importance of consistency versus ambiguity. It was found in [9] that an even better measure is given by the inner product

$$\vec{p}_i \cdot \vec{q}_i \qquad (5)$$

The global measure is then an average over the set of units of the local measures. We can thus define

$$C = \sum_{\substack{\text{all units} \\ u_i}} \vec{p}_i \cdot \vec{q}_i \tag{6}$$

as a global criterion over the set of units that measures the consistency and ambiguity of the labeling.

The labeling problem is now equivalent to the following:

$\mathcal{P}$ { given an initial labeling $\vec{p}_i^{(0)}$, find the local maximum of criterion C closest to the original labeling subject to the constraint that vectors $\vec{p}_i$ are probability vectors.

This is typically a constrained optimization problem which as shown in [7,8] can be efficiently solved by using steepest descent techniques. In particular, we can attach to every unit u_i a local gradient vector $\vec{g}_i = \frac{\partial C}{\partial \vec{p}_i}$ and define an iteration scheme as:

$$\vec{p}_i^{(n+1)} = \vec{p}_i^{(n)} + \rho_n P\{\vec{g}_i^{(n)}\} \quad n \geq 0 \tag{7}$$

where ρ_n is a positive number and P a linear projection operator. For a description of P see [7].

It can be easily shown that:

$$q_i(k) = q_i(k) + \sum_{\substack{\text{neighbors} \\ \text{of } u_i}} \frac{1}{D_j} \sum_{\ell \text{ in } W_j} C(j,\ell,i,k)\,(p_j(\ell) - \vec{p}_j \cdot \vec{q}_j) \qquad \text{for } k=1,\ldots,L_i \tag{8}$$

where

$$D_j = \sum_{\ell=1}^{L_j} Q_j(\ell) \qquad (9)$$

The first term $q_i(k)$ in Eq. (6) corresponds to the simple maximization of the product $\vec{p}_i \cdot \vec{q}_i$ in the global criterion C, whereas the second term corresponds to the coupling between units through the compatibility relations. The algorithm defined by Eq. (5) will allow us to evolve from the initial stochastic labeling toward a less ambiguous and more consistent labeling.

Least Commitment Versus Speed, Multiple Matches

The task of matching a model with a symbolic representation of an image obtained by a automatic segmentation procedure presents the important characteristic that matches may not be unique. For example, if a highway in the image has been separated by the segmentation procedure into several disconnected pieces, then each piece is a potential correct match for the node "highway" in the model.

One possible way of handling this problem would be to relax the constraint that vectors $\vec{p}_i$ are probability vectors and interpret them as confidence vectors for which every component may vary between 0 and 1. This would have the benefit of delaying any firm commitment as much as possible, but would also have a general tendency to slow the convergence of the iterative scheme. This is why we looked for an alternative between an increase in speed (and therefore of the probability of making errors) and the principle of least commitment.

We therefore introduced the notion of a Macro-iteration composed of several of the iterations defined by Eq. (7) (Micro-iterations) after which decisions are made to assign names to units based upon the comparison of the components of the vectors $\vec{p}_i$ to a threshold (usually 80%). If one component is larger than the threshold then it is set

equal to 1 and unit u_i is considered to be assigned the corresponding name n_k.

The process is then reinitialized by computing new initial probabilities $\vec{p}_i^{(0)}$. For units u_i which have been assigned names these are not actually probability vectors any more. The sum of their components adds up to k_i+1 where k_i is the number of assigned names. Since relations are used to compute initial probabilities only when units are assigned (otherwise only features are used), this also has the advantage of improving the original estimates.

In the optimization problem ($\mathcal{P}$) the constraints are now:

$$\begin{cases} k_i \leq \sum_{k=1}^{L_i} p_i(k) \leq k_i+1 \\ \\ p_i(\ell) = 1 \text{ for all names } \ell \text{ which have already been assigned to unit } u_i \end{cases}$$

That is, we do not allow the confidence value for already assigned names to be changed from 1.

Comparison With Other Approaches And Experimental Results

We compared our approach with results from two other techniques: the relaxation procedure originally introduced by Rosenfeld, Hummel and Zucker [10] (algorithm RHZ) and the sequential matching procedure developed by Nevatia and Price [6] (algorithm NP). The nonlinear iterative updating formula of algorithm RHZ can be expressed as

$$p_i^{(n+1}(k) = \frac{p_i^{(n)}(k)Q_i(k)}{\sum_{\ell=1}^{L_i} p_i^{(n)}(\ell)Q_i(\ell)} \qquad (10)$$

where the Q_i's are computed in the same way as in Eq. (2). On this particular proble, algorithm RHZ has a tendency to converge toward ambiguous solutions. The reason is, of course that this algorithm does not take into account completely the coupling between units as reflected by the c(i,k,j,ℓ)'s and fails to account for the notion of ambiguity. As shown in Eqs. (6) and (8) this is not the case with the algorithm described in this paper.

The basic comparison procedure used here is the same as used in algorithm NP, so that many of the results are identical. The major problems with the sequential method are the lack of a consistent method for the handling of multiple assignments and errors which are caused by the order in which the objects are selected for matching. For example, if a pair of identical objects are near each other, the sequential procedure can easily find the wrong assignment for the first object, thus forcing the second to also be incorrect, due to the limit of assigning one name to only one unit (units may be assigned several names). Since the relaxation procedure is considering pairs of objects, the correct assignments advance to the top, if there are relations connecting the two objects.

Results

We have applied this procedure to several different scenes or portions of scenes and will present results from two of these scenes. The first is a high altitude aerial image with a few major regions (a city and rural areas) and a number of linear features (a major highway, river channels, and roads along the channels), see Fig. 1. The second is a closer view (of a different area), with 2 roads included in the model. The roads are described in sections because of the tendency of the linear feature extraction procedure to break long linear features into shorter segments, see Fig. 2.

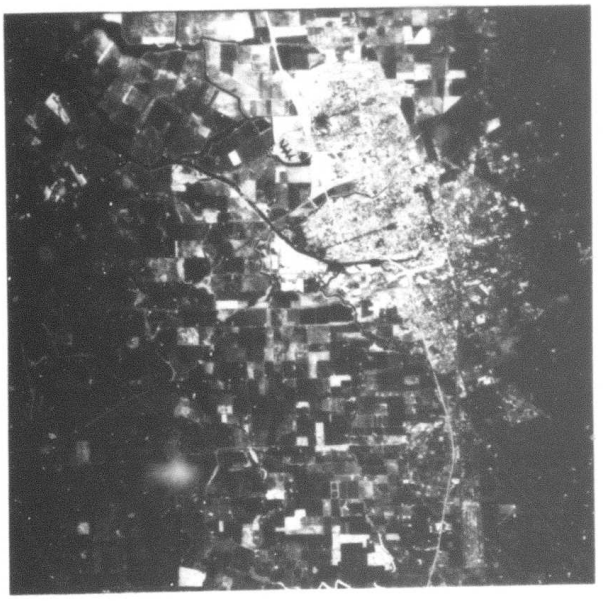


Fig. 1 Original high altitude image of Stockton, California area.

Fig. 2 Original low altitude image of Fort Belvoir, Virginia area.

To illustrate how the relaxation procedure operates, Fig. 3 is a graph of the probabilities of various assignments for one object through a series of macro iterations.

Figs. 4 and 5 show another aspect of the operation of the relaxation process. The 15 most likely assignments for one model object are shown with the most likely one labeled 1 and least likely labeled 15. The first picture is the initial likelihoods based primarily on feature values, the succeeding pictures are the likely assignments after 2 and 5 (micro) iterations. Figs. 6 and 7 show the final results, with the objects outlined and labeled. Many of the labels overlap since adjacent linear features are being presented.

References

1. S. Rubin, "The ARGOS Image Understanding System," Ph.D. thesis, Computer Science Department, Carnegie-Mellon U., Pittsburgh, PA, 1978.

2. D. Smith, "Search Strategies for the ARGOS Image Understanding System," in Proc. Image Understanding Workshop, Los Angeles, Ca. Nov. 1979, pp. 42-46.

3. K. Price and R. Reddy, "Matching Segments of Images," IEEE Trans-PAMI Vol. 1, Jan, 1979, pp. 110-116.

4. R. Ohlander, K. Price and R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," Comp. Graphics and Image Processing, Vol. 8, pp. 313-333, 1978.

5. R. Nevatia, and K.R. Babu, "Linear Feature Extraction and Description," to appear in Computer Graphics and Image Processing.

6. R. Nevatia and K. Price, "Locating Structures in Aerial Images," submitted for publication.

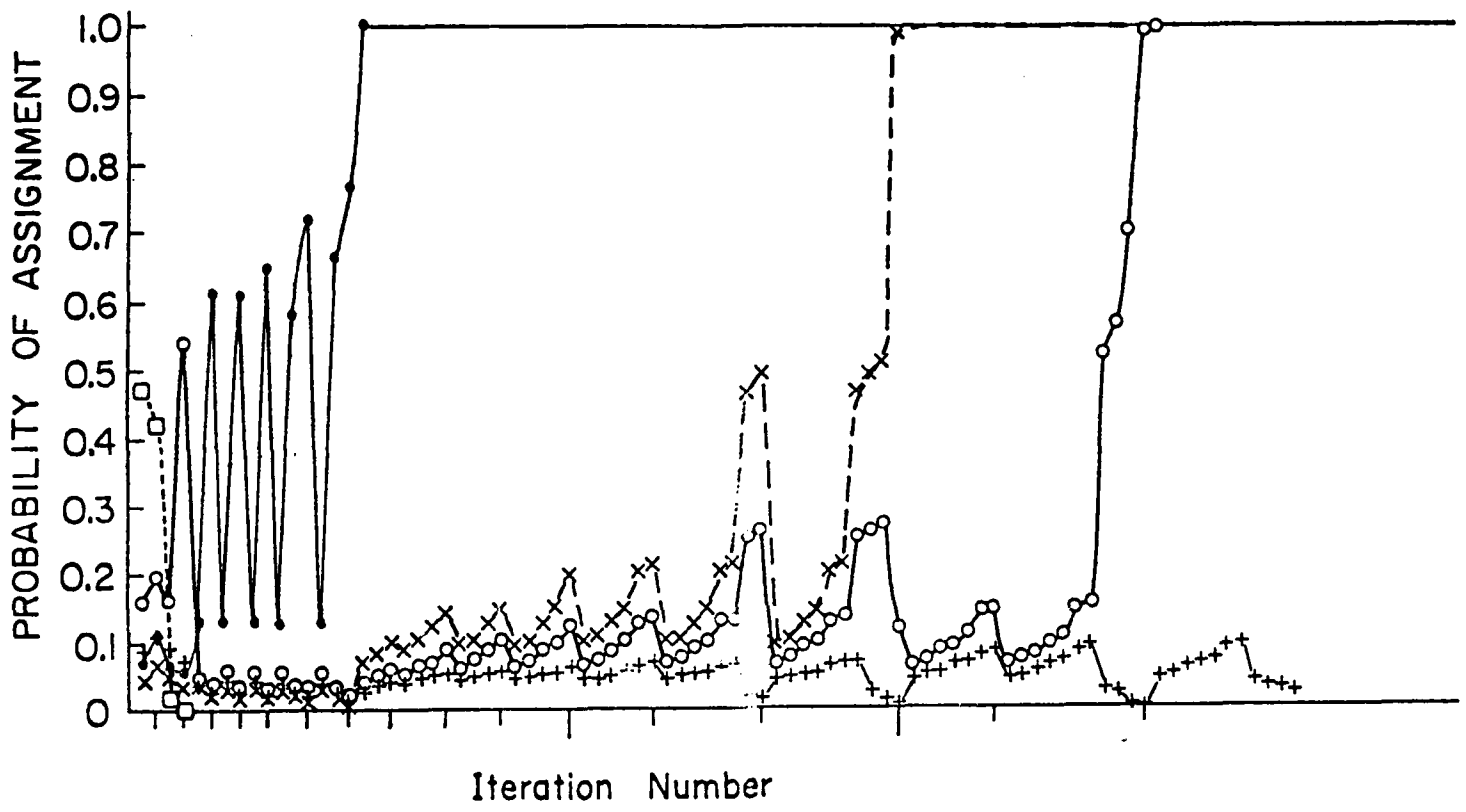
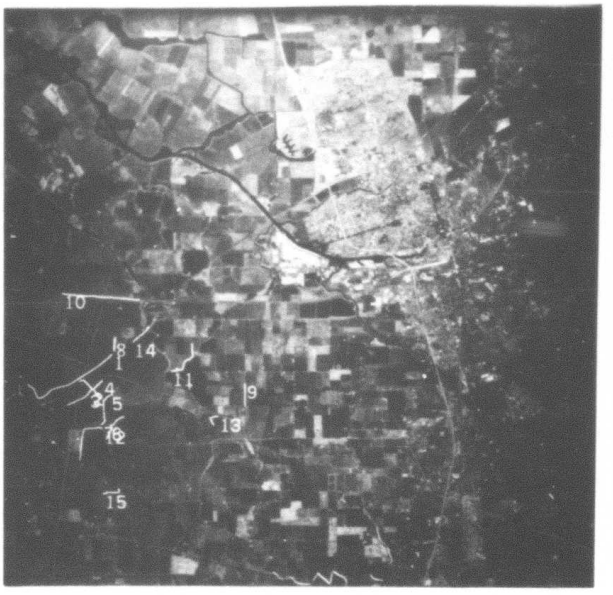


Fig. 3. Graph of the match likelihoods for several image segments to be assigned to one model element. The model element being considered is labeled "SOUTH-HIGHWAY" in the final results of Fig. 6. The tick marks along the horizontal axis indicate each Macro iteration. The three segments which rise to 1.0 are all correct assignments. Note also that the initial best assignments is not considered after the firm assignment made on the second macro iteration.

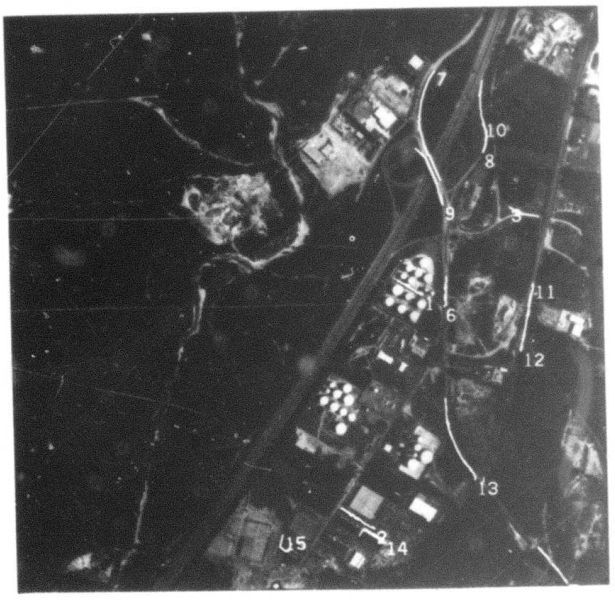
(a)



(b)

(c)

Fig. 4 Sequence of 15 most likely labels for a river area. a) Ordered by initial probabilities. The correct assignment is also the most likely. b) Likely assignments after 2 iterations.c) Likely assignments after 4 iterations (now only 10 with nonzero likelihood).



(a)

(b)

(c)

Fig. 5 Sequence of 15 most likely labels for a road segment. a) Ordered by initial probabilities based primarily on feature values. The correct label is the sixth most likely label. b) Likely assignments after 3 iterations. c) Likely assignments after 6 iterations.

Fig. 6 Final assignments for Stockton area.

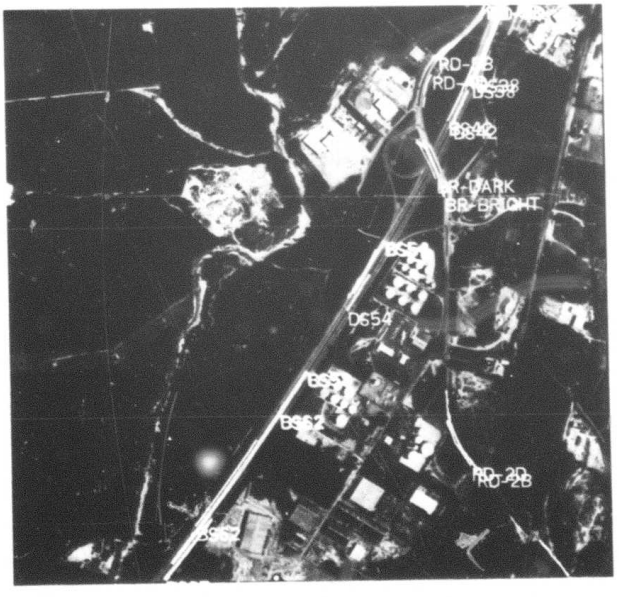


Fig. 7 Final assignments for road segments of Ft. Belvoir area.

7. O.D. Faugeras and M. Berthod, Scene Labeling: An Optimization Approach," Proceeding of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, pp. 318-396, Chicago, August 6-8, 1980.

8. O.D. Faugeras and M. Berthod, "Improving Consistency and Reducing Ambiguity in Stochastic Labeling: An Optimization Approach," submitted to the IEEE Trans. on Pattern Analysis and Machine Intelligence, November 1979.

9. M. Berthod and O.D. Faugeras, "Using Context in the Global Recognition of a Set of Objects: An Optimization Approach,"

10. A. Rosenfeld, R.A. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. on Syst., Man, and Cybern. SMC-6, No. 6, pp. 420-453, June 1976.

2.2 Decomposition and Decentralization Techniques in Relaxation Labeling

O.D. Faugeras

Introduction

One of the most important tasks in Image Analysis is to assign names to objects present in the scene. This semantic analysis usually happens after some symbolic analysis like segmentation and shape analysis has been performed but it does not have to. Low level vision can also be considered as semantic analysis when for example we attempt to decide whether an edge is present or not at some point in

an image (names are here edge/noedge) or when we try to segment a picture into objects and background (names are here objects/background). Usually the task of assigning names to objects is very difficult on the basis of only features measured for that object. This is typically a Pattern Recognition problem and errors are likely to be made for reasons such as noisy data and inadequate feature extraction. One of the key ideas in getting around that problem is to delay any firm commitment until all the available contextual information has been used.

Waltz [1] was one of the first to use the notion of constraints induced by a world model to solve the problem of analyzing a scene made up of complex polyhedra. His ideas were generalized by Rosenfeld et al. [2] who proposed a completely parallel version of Waltz labeling algorithm. Davis and Rosenfeld presented some applications of the so-called "discrete" relaxation techniques to template matching [3], shape matching [4] and waveform parsing [5]. Haralick [6] introduced a general framework, the theory of arrangements, in which the solution to some scene analysis tasks can be found by constructing a homomorphism from one arrangement to another. To help determine all these homomorphisms, he generalizes to N dimensions the Waltz filtering, the discrete scene labeling relaxation and the network consistency relation [7].

Barrow and Tenenbaum [8] and Rosenfeld, Hummel and Zucker [2] introduced the idea of probabilistic or stochastic labeling where we not only know possible and impossible classes for every object but also a measure of their likelihood. Similarly the world model is now described continuously in terms of compatibility coefficients or conditional probabilities between labels. The nonlinear algorithm proposed in [2] has been used in different applications, edge and line enhancement [9,10,11], clustering [11,12], stereo vision and movement detection [13]. Similar ideas have been used by Marr and Poggio [14]. Theoretical analysis of the convergence and stability properties of this algorithm has proven to be difficult as shown by Zucker

[15,16,17]. Faugeras and Berthod [18,19,20] have proposed a different approach based upon explicit use of compatibility and ambiguity to define a global criterion on the set of objects. They showed that the criterion could be minimized using projected gradient techniques. A similar idea has been proposed by Ullman [21]. We will not proceed to describe further the problem of stochastic labeling viewed as an optimization task.

Stochastic labeling as a problem in Optimization

Definition of criterion:

Formally, we are given a set of N objects $a_1, a_2, \ldots, a_N$ which fall into L classes $\lambda_1, \ldots, \lambda_L$. To every object a_i we assign a probability noted $p_i(\lambda_k)$ to belong to class λ_k. This is conveniently represented as a vector $\vec{p}_i = [p_i(\lambda_1), \ldots, p_i(\lambda_L)]^T$. Objects are also related to one-another: we will denote by $V_i(\lambda_k)$ the set of objects related to object a_i for label λ_k. This means that for every object a_j in $V_i(\lambda_k)$ we are given the conditional probabilities $p_{ij}(\lambda_k|\lambda_\ell)$ that object a_i belongs to class λ_k given that object a_j belongs to class λ_ℓ, $\ell=1,\ldots,L$. At this point we would like to stress the fact that the vectors $\vec{p}_i$ are the results of measurements taken on the data to be analyzed and that the $p_{ij}(\lambda_k|\lambda_\ell)$'s constitute our a priori contextual information or world model. Due to noisy data, vectors $\vec{p}_i$ are not always compatible with respect to the contextual information and ambiguous, that is, in general we do not have

$$p_i(\lambda_k) = \sum_{\ell=1}^{L} p_{ij}(\lambda_k|\lambda_\ell) p_j(\lambda_\ell) \qquad (1)$$

and probability vectors $\vec{p}_i$ are not unit vectors $[0,\ldots 0,1,0\ldots 0]^T$. From these two facts stems the need for designing algorithms that use contextual information and the notion of ambiguity to modify the probability for every object in such a way that consistency is

increased and ambiguity decreased.

The approach that we suggested [18,19,20] was based upon the minimization of a criterion defined over the set of objects. The class of criteria that we proposed was meant to be a global measure of the amount of consistency and ambiguity present in the stochastic labeling. Local consistency is measured as the norm of the difference of the probability vector $\vec{p}_i$ for object a_i and a consistency vector $\vec{q}_i=[q_i(\lambda_1),\ldots,q_i(\lambda_L)]^T$ whose coordinates are some normalized averages of what we would expect the probabilities of labels for object a_i to be when taking into account both the measurements at objects related to a_i and the contextual information. More precisely, we have:

$$q_i(\lambda_k) = \frac{Q_i(\lambda_k)}{\sum_{\ell=1}^{L} Q_i(\lambda_\ell)} \tag{2}$$

and

$$Q_i(\lambda_k) = \left(\frac{1}{|V_i(\lambda_k)|} \sum_{j\in V_i(\lambda_k)} s_{ijk}^r\right)^{\frac{1}{r}} \qquad 0 < r < \infty \tag{3}$$

The positive numbers s_{ijk} are given by:

$$s_{ijk} = \alpha_{ijk} \sum_{\ell=1}^{L} p_{ij}(\lambda_k|\lambda_\ell) p_j(\lambda_\ell) \tag{4}$$

The α_{ijk}'s are positive and weight the importance we attribute to the different objects a_j in $V_i(\lambda_k)$ in evaluating the consistency of label

λ_k for object a_i (index j) as well as the importance we attribute to that particular label in ascertaining the total consistency for object a_i (index k). As was previously mentioned, this quantity is measured by the vector norm

$$\|\vec{p}_i - \vec{q}_i\|_\alpha = \left(\sum_{k=1}^{L} |p_i(\lambda_k) - q_i(\lambda_k)|^\alpha \right)^{\frac{1}{\alpha}} \qquad 0 < \alpha < \infty \tag{5}$$

Global consistency is then measured by an average taken over the set of objects of the local consistency measures, for example the arithmetic mean:

$$C = \frac{1}{N} \sum_{i=1}^{N} \|\vec{p}_i - \vec{q}_i\|_\alpha \tag{6}$$

In the same spirit, local ambiguity can be measured [18,19,20] by the entropy of the probability vector $\vec{p}_i$ as defined by

$$H_i = \sum_{k=1}^{L} p_i(\lambda_k)(1 - p_i(\lambda_k)) = 1 - \|\vec{p}_i\|_2^2 \tag{7}$$

Global ambiguity is then measured by the average of the local ambiguity measures:

$$A = 1 - \frac{1}{N} \sum_{i=1}^{N} \|\vec{p}_i\|_2^2 \tag{8}$$

The total criterion J is then defined as

$$J = \omega C + (1-\omega) A \tag{9}$$

where coefficient ω which varies between 0 and 1 weights the relative importance we attribute to consistency versus ambiguity.

Computational aspects

Let us denote by $\underline{v}$ the vector of $R^M=\underbrace{R^L x \ldots x R^L}_{N \text{ times}}$ $(M=NL)$ equal to $(\vec{p}_1,\vec{p}_2,\ldots,\vec{p}_N)$. We can define the stochastic labeling problem as follows:

given an initial stochastic labeling $\vec{v}^0$ find the stochastic labeling $\vec{u}$ that corresponds to the local minimum of criterion $J(\vec{v})$ which is closest to $\vec{v}^0$ subject to the constraint that if $\vec{u}=(\vec{p}_1,\ldots,\vec{p}_N)$ then $\vec{p}_i$ is a probability vector for $i=1,\ldots,N$

The fact that we are not requiring to find the absolute minimum of J corresponds to the desire to evolve toward a stochastic labeling which depends on the initial measurements. The contrary would imply that the final labeling would be independent of the original set of objects and would depend solely on the world model!

The optimization problem may in practice be of a very large dimensionality if the number of objects and possible labels is not kept within reasonable bounds. Unfortunately, for many practical problems this is not possible and we have to find means of getting around the "curse of dimensionality". In [20] we show that a large amount of parallelism can be introduced in the minimization process: if we attach to every object a_i a processor r_i connected only to processors r_j attached to objects a_j related to a_i, then the global criterion J can be minimized by having processors r_i perform simple operations mostly in parallel while a simple sequential communication process allows them to work toward the final goal in a coordinated fashion.

Another set of related ideas can be found in the so-called Decomposition and Decentralization techniques which have been developed to solve similar problems in Economy, Numerical Analysis, Systems Theory and Optimal Control [22,23,24,25]. Decomposition techniques proceed from an algorithmic standpoint: we are confronted with a problem of large dimensionality and try to substitute for it a sequence of problems of smaller dimensionalities. Decentralization techniques take a different viewpoint: we are confronted with a global problem and have at our disposal P decision centers. The question that we ask ourselves is whether it is possible to solve the global problem while letting the decision centers solve only local problems.

Our purpose here is to show that the structure of our criterion allows us to develop both types of techniques. In order to do that we need to develop a few notations.

Grouping objects into sets

Let Ω_k, k=1,...,P be nonoverlapping sets of objects, that is

$$\Omega_k \cap \Omega_\ell = \emptyset \qquad \text{if } k \neq \ell \tag{10}$$

For ever object a_i, i=1,...,N we will denote by V_i the set of all objects a_j related to it for some labels λ_k:

$$V_i = \bigcup_{k=1}^{L} V_i(\lambda_k) \tag{11}$$

Two sets Ω_k and Ω_ℓ can "interact" if for some object a_i in Ω_k, some of its neighbors are in Ω_ℓ that is if $V \cap \Omega_\ell \neq 0$. We will then define $\bar{\Omega}_k$ to be

$$\bar{\Omega}_k = \bigcup_{\substack{\ell=1 \\ \ell \neq k}}^{P} \left\{ \begin{array}{l} \text{all objects in } \Omega_\ell \text{ which are neighbors} \\ \text{of some objects in } \Omega_k \end{array} \right\} \cup \Omega_k \qquad (12)$$

and N_k to be the number of objects in $\bar{\Omega}_k$. Notice that we may now have $\bar{\Omega}_k \cap \bar{\Omega}_\ell \neq 0$ for some pairs (k,ℓ) with $k \neq \ell$. Let us call $\sum_{\min(k,\ell)\max(k,\ell)}$ the previous intersection and $N_{\min(k,\ell)\max(k,\ell)}$ the number of objects in it. Given a subset Ω_k of objects we will denote by $\vec{v}|_{\Omega_k}$ the restriction of the vector $\vec{v}=(\vec{p}_1,\ldots,\vec{p}_N)$ to objects in Ω_k.

Let us now take a closer look at the structure of criterion J. Clearly it can be rewritten as a sum:

$$J(\vec{v}) = \sum_{k=1}^{N} J_k(\vec{v}) \qquad (13)$$

where, according to equations (6), (8) and (9)

$$J_k(\vec{v}) = \frac{1}{N}\left(\omega \|\vec{p}_k - \vec{q}_k\|_\alpha + (1-\omega)\left(1 - \|\vec{p}_k\|_2^2\right)\right) \qquad (14)$$

From the definition of vector $\vec{q}_k$, it is seen that criterion $J_k(\vec{v})$ depends solely upon vectors $\vec{p}_j$ for objects a_j in V_k or to put it another way that $J_k(\vec{v}) \equiv J_k(\vec{v}_k)$ where $\vec{v}_k$ is the restriction of $\vec{v}$ to objects in $\bar{\Omega}_k^{(0)}$ ($\Omega_k^{(0)}=\{a_k\}$. This decomposition of criterion J as a sum of N criteria corresponds to the partition of the set of objects into the subsets $\Omega_k^{(0)}$, k=1,...,N. More generally, to every partition Ω_k, k=1,...,P of the set of objects we can associate a decomposition of criterion J as a sum of P criteria:

$$J(\vec{v}) = \sum_{k=1}^{P} C_k(\vec{v}) \tag{15}$$

where

$$C_k(\vec{v}) = \sum_{i \text{ in } \Omega_k} J_i(\vec{v}) \equiv C_k(\vec{v}_k) \tag{16}$$

and

$$\vec{v}_k = \vec{v}\Big|_{\Omega_k} \tag{17}$$

Decomposition techniques

Our problem is to minimize a criterion $J(\vec{v})$ given by:

$$J(\vec{v}) = \sum_{k=1}^{P} C_k(\vec{v}) \equiv \sum_{k=1}^{P} C_k(\vec{v}_k) \tag{18}$$

where $\vec{v}=(\vec{p}_1,\ldots,\vec{p}_N)$ is subject to the constraint that $\vec{p}_i$ is a probability vector for i=1,...,N and

$$C_k(\vec{v}) = \sum_{a_i \text{ in } \Omega_k} J_i(\vec{v}) \tag{19}$$

The set of constraints can be expressed as

$$K = \bigcap_{k=1}^{P} K_k \tag{20}$$

$K_k = \{\vec{v} \mid \vec{v} = (\vec{p}_1, \ldots, \vec{p}_N)$ such that for all objects a_i in $\overline{\Omega}_k \vec{p}_i$ is a probability vector$\}$.

The decomposition of K according to (20) corresponds to a splitting of constraints: the idea is to minimize a criterion with a lesser number of constraints.

Sequential algorithm

Let $\tau_0, \tau_1, \ldots$ be a sequence of strictly positive numbers we define a family of vectors $\vec{u}^{n+\frac{i}{P}}$ i=1,...,P, n=0,1,... starting from any $\vec{u}^{(0)}$ in K and assuming that we have computed $\vec{u}^0, \ldots, \vec{u}^{n+\frac{k-1}{P}}$, then $\vec{u}^{n+\frac{k}{P}}$ is the element of K_k that minimizes

$$C_k(\vec{v}) + \frac{1}{2\tau_n} \left\| \vec{v} - \vec{u}^{n+\frac{k-1}{P}} \right\|^2 \tag{21}$$

We can associate with the sequence of vectors $\vec{u}^{n+\frac{k}{P}}$ the sequence $\vec{\omega}^{N+\frac{k}{P}}$ given by

$$\vec{\omega}^{N+\frac{k}{P}} = \frac{1}{\sigma_N} \sum_{n=0}^{N} \tau_n \vec{u}^{n+\frac{k}{P}} \tag{22}$$

where σ_N is the sum of the numbers τ_n, n=0,...,N.

It can be shown [25] that the limit when N goes to infinity of $\vec{\omega}^{N+\frac{k}{P}}$ is a local minimum of criterion J for k=1,...,P. The algorithm then goes as follows:

Algorithm DS:

step 1. (Initialization) start with $\vec{u}^0$ in K, set n equal to 0

step 2. (Iteration initialization) set k equal to 1

step 3. (Solve problem) find the minimum $\vec{u}^{n+\frac{k}{P}}$ in K_k closest to $\vec{u}^{n+\frac{k-1}{P}}$ of criterion

$$C_k(\vec{v}) + \frac{1}{2\tau_n} \| \vec{v} - \vec{u}^{n+\frac{k-1}{P}} \|^2 \tag{23}$$

step 4. (Test for end of loop) set k equal to k+1. If k less or equal P go to step 3.

step 5. (Test for convergence) if converged stop else set n equal to n+1 and go to step 2.

If we stopped for n=N the result is

$$\vec{\omega}^{N+\frac{k}{P}} = \frac{1}{\sigma_N} \sum_{n=0}^{N} \tau_n \vec{u}^{n+\frac{k}{P}} \tag{24}$$

for k=1,...,P. In practice of course, since we stop after a finite number of iterations, vectors $\vec{\omega}^{N+\frac{k}{P}}$ are likely to be different for k=1,...,P. They should be nonetheless close to one another and we can either choose one of them as the final result or their average

$$\vec{\bar{\omega}}^{N} = \frac{1}{P} \sum_{k=1}^{P} \vec{\omega}^{N+\frac{k}{P}} \tag{25}$$

Parallel algorithm

Again we start with a vector $\vec{u}^0$ in K and supposing that we know $\vec{u}^0,\ldots,\vec{u}^n$, we define $\vec{u}^{n+\frac{k}{P+1}}$ for k=1,...,P as a local minimum of

$$C_k(\vec{v}) = \frac{1}{2\tau_n} \|\vec{v}-\vec{u}^n\|^2 \tag{26}$$

for $\vec{v}$ in $K_k \cdot \vec{u}^{n+1}$ is then defined as

$$\vec{u}^{n+1} = \frac{1}{P} \sum_{k=1}^{P} \vec{u}^{n+\frac{k}{P+1}} \tag{27}$$

Just as for the sequential algorithm, it is possible to show [25] that the sequence of vectors $\vec{\omega}^{N+\frac{k}{P+1}}$ defined as

$$\vec{\omega}^{N+\frac{k}{P+1}} = \frac{1}{\sigma_N} \sum_{n=0}^{N} \tau_n \vec{u}^{n+\frac{k}{P+1}} \tag{28}$$

has a limit when N goes to infinity and that this limit is a local minimum of criterion J for k = 1,...,P. The algorithm then goes as follows:

Algorithm DP:

step 1. (Initialization) start with $\vec{u}^0$ in K and set n equal to 0

step 2. (Solve problems in parallel) find the minimum $\vec{u}^{n+\frac{k}{P+1}}$ in K_k closest to $\vec{u}^n$ of criterion

$$C_k(\vec{v}) + \frac{1}{2\tau_n} \|\vec{v}-\vec{u}^n\|^2 \tag{29}$$

for k = 1,...,P

Step 3. (Test for convergence) if converged stop else set $\vec{u}^{n+1}$ equal to $\frac{1}{P} \sum_{k=1}^{P} \vec{u}^{n+\frac{k}{P+1}}$, n equal to n+1 and go to step 2.

If we stopped at N the result is

$$\vec{\omega}^{N+\frac{k}{P+1}} = \frac{1}{\sigma_N} \sum_{n=0}^{N} \tau_n \vec{u}^{n+\frac{k}{P+1}} \qquad (30)$$

for k = 1,...,P. Just as in the case of the sequential algorithm since we stop after a finite number of iterations, vectors $\vec{\omega}^{N+\frac{k}{P+1}}$ are likely to be different for k = 1,...,P. The same remedy can be used.

Simple case analysis

Let us consider a set $\Omega = \{a_1, a_2, a_3, a_4\}$ of four objects related as in figure 1, that is to say

$$V_i = \{a_{i-1}, a_{i+1}\} \qquad i = 1,...,4 \qquad \text{where} \quad \begin{matrix} a_0 = a_4 \\ a_5 = a_1 \end{matrix} \qquad \text{and}$$

Let us now choose for example

$$\Omega_i = \{a_i\} \qquad i = 1,...,4$$

We thus have $\overline{\Omega}_i = \{a_i\} \cup V_i$ for i = 1,...,4 and

$$\vec{v}_1 = (\vec{p}_1, \vec{p}_2, \vec{p}_4)$$

$$\vec{v}_2 = (\vec{p}_1, \vec{p}_2, \vec{p}_3)$$

$$\vec{v}_3 = (\vec{p}_2, \vec{p}_3, \vec{p}_4)$$

$$\vec{v}_4 = (\vec{p}_1, \vec{p}_3, \vec{p}_4)$$

We also have:

$$C_1(\vec{v}) \equiv C_1(\vec{v}_1) = \frac{\omega}{4}\|\vec{p}_1-\vec{q}_1\|^2 + \frac{1-\omega}{4}(1-\|\vec{p}_1\|^2) \tag{31}$$

where according to figure 1 and equations (2-4), vector $\vec{q}_1$ depends solely upon vectors $\vec{p}_2$ and $\vec{p}_4$. Starting with initial conditions $\vec{u}^0 \equiv (\vec{p}_1^0, \vec{p}_2^0, \vec{p}_3^0, \vec{p}_4^0)$, in the case of the sequential algorithm we minimize first

$$C_1(\vec{v}) + \frac{1}{2\tau_0}(\|\vec{p}_1-\vec{p}_1^0\|^2 + \|\vec{p}_2-\vec{p}_2^0\|^2 + \|\vec{p}_3-\vec{p}_3^0\|^2 + \|\vec{p}_4-\vec{p}_4^0\|^2) \tag{32}$$

since $C_1(\vec{v})$ does not depend upon $\vec{p}$ the result will be

$$\vec{u}^{0+\frac{1}{4}} = (\vec{p}_1^{\frac{1}{4}}, \vec{p}_2^{\frac{1}{4}}, \vec{p}_3^{0}, \vec{p}_4^{\frac{1}{4}}) \tag{33}$$

Similarly we will have:

$$\vec{u}^{0+\frac{2}{4}} = (\vec{p}_1^{\frac{2}{4}}, \vec{p}_2^{\frac{2}{4}}, \vec{p}_3^{\frac{2}{4}}, \vec{p}_4^{\frac{1}{4}}) \tag{34}$$

In the case of the parallel algorithm we first compute $\vec{u}^{0+\frac{1}{5}} = \vec{u}^{0+\frac{1}{4}}$ but we then have

$$\vec{u}^{0+\frac{2}{5}} = (\vec{p}_1^{\frac{2}{5}}, \vec{p}_2^{\frac{2}{5}}, \vec{p}_3^{\frac{2}{5}}, \vec{p}_4^{0}) \tag{35}$$

which, in general is different from $\vec{u}^{0+\frac{2}{4}}$.

Thus the two algorithms are not trivially equivalent in the sense that at every iteration they compute essentially the same thing in different ways. They are nonetheless equivalent in a deeper sense that in the limit they converge toward the same answer. In practice it would be interesting to know whether or not one algorithm converges faster than the other.

We would like to point out that this simple example is only meant to help the reader get a more concrete feeling for the concepts developed in the previous sections. Because of the small number of

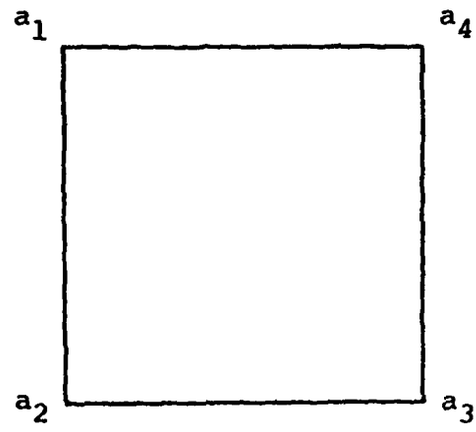


Figure 1. Simple example with four objects

objects in the initial problem we replaced a problem of dimension 4L with 4 problems of dimension 3L. Of course the gain in dimensionality can be a lot more for larger size problems as encountered in practice. The question of how to optimally choose numbers τ_n's to speed up convergence is largely unsolved although some preliminary results have been obtained [26].

Price decentralization techniques

General Algorithm:

The idea here is to let the decision center solve local problems and coordinate them in such a way that the overall result is the minimization of the global criterion J. When minimizing $C_k(\vec{v}_k)$ over $\overline{\Omega}_k$ the k^{th} decision center is going to compute vectors $\vec{p}_{kj}$ for objects a_j in regions Ω_ℓ adjacent to region Ω_k and there is no reason for having $\vec{p}_{kj}=\vec{p}_{\ell j}$, $\vec{p}_{\ell j}$ being the result of the minimization of $C_\ell(\vec{v}_\ell)$ by the ℓ^{th} decision center over $\overline{\Omega}_\ell$. Thus the need for coordination which is achieved by attributing a price to the discrepancy between $\vec{p}_{kj}$ and $\vec{p}_{\ell j}$.

Formally, we need to consider intersections $\sum_{k\ell}$ between sets $\overline{\Omega}_k$ and $\overline{\Omega}_\ell$ $(k<\ell)$ and define

$$\Psi = \prod_{k=1}^{P-1} \prod_{\ell=k+1}^{P} (R^L)^{N_{k\ell}} \tag{36}$$

For each set $\overline{\Omega}_k$ we then define a linear application B_k of $(R^L)^{N_k}$ into Ψ as:

$$B_k(\vec{v}_k) = \begin{cases} \vec{v}_k \Big|_{\Sigma_{kj}} & j > k \\ -\vec{v}_k \Big|_{\Sigma_{ik}} & i < k \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

where, as usual, $\vec{v}_k\Big|_{\Sigma_{ij}}$ is the restriction of v_k to objects in $\sum_{ij}$. The global problem

$$\min_{\vec{v}} J(\vec{v}) \tag{38}$$

is now equivalent to the problem:

$$\begin{cases} \min\limits_{\vec{v}_1,\ldots,\vec{v}_P} \sum\limits_{k=1}^{P} C_k(\vec{v}_k) \\ \text{subject to } \sum\limits_{k=1}^{P} B_k(\vec{v}_k) = \vec{0} \end{cases} \tag{39}$$

which in turn is equivalent to finding a saddle point of the functional (Lagrangian):

$$L(\vec{v}_1,\ldots,\vec{v}_P,\vec{r}) = \sum_{k=1}^{P} C_k(\vec{v}_k) + \vec{r}\cdot\left(\sum_{k=1}^{P} B_k(\vec{v}_k)\right) \tag{40}$$

where the price vector (Lagrange multiplier) $\vec{r}$ is a vector of ψ and "." denotes the inner product.

The algorithm then goes as follows:

Algorithm PD:

step 1. (Initialization) set ℓ equal to 1, choose a price vector $\vec{r}(1)$

step 2. (Local problems) solve the P "local" problems

$$\min_{\vec{v}_k} C_k(\vec{v}_k)+\vec{r}^{(\ell)}\cdot B_k(\vec{v}_k) \tag{41}$$

the results are $\vec{v}_k^{(\ell)}$, k=1,...,P.

step 3. (Test for convergence) if converged stop else set ℓ equal to ℓ+1 and go to step 4.

step 4. (Update the price) $\vec{r}^{(\ell+1)}$ is obtained by

$$\vec{r}^{(\ell+1)} = \vec{r}^{(\ell)}+\rho_\ell \sum_{k=1}^{P} B_k(\vec{v}_k^{(\ell)}) \tag{42}$$

where ρ_ℓ is an acceleration parameter, go to step 2.

Just as in the case of the Decomposition techniques we have to minimize a sequence of slightly perturbed versions of criteria $C_k(\vec{v}_k)$ for k=1,...,P. Two important differences must be pointed out: in the case of the price Decentralization technique, the perturbation term depends solely upon $\vec{v}_k$, rather than $\vec{v}$ for Decomposition techniques and is a linear function of $\vec{v}_k$ rather than a quadratic one.

Simple case analysis

Just as in the case of Decomposition Techniques, it may be helpful at this point to examine more closely a concrete example. We chose the same as in a previous section. We will need this line to distinguish between the components of the different vectors $\vec{v}_k = \vec{v}\big|_{\bar{\Omega}_k}$, for k=1,2,3,4. We will therefore write:

$$\vec{v}_1 = (\vec{p}_{11}, \vec{p}_{12}, \vec{p}_{14})$$
$$\vec{v}_2 = (\vec{p}_{21}, \vec{p}_{22}, \vec{p}_{23})$$
$$\vec{v}_3 = (\vec{p}_{32}, \vec{p}_{33}, \vec{p}_{34})$$
$$\vec{v}_4 = (\vec{p}_{31}, \vec{p}_{33}, \vec{p}_{34})$$

We also have:

$$C_1(\vec{v}) \equiv C_1(\vec{v}_1) = \frac{\omega}{4} \| \vec{p}_{11} - \vec{q}_{11} \|^2 + \frac{(1-\omega)}{4} (1 - \| \vec{p}_{11} \|^2) \qquad (43)$$

where, according to figure 1 and equation (24), vector $\vec{q}_{11}$ depends only on vectors $\vec{p}_{12}$ and $\vec{p}_{14}$. We must also consider

$$\sum_{ij} = \bar{\Omega}_i \cap \bar{\Omega}_j \qquad i,j = 1,2,3,4 \qquad i < j \qquad (44)$$

According to a previous section, we have:

$$\sum_{12} = \{a_1, a_2\} \qquad \sum_{13} = \{a_2, a_4\} \qquad \sum_{14} = \{a_1, a_4\}$$

$$\sum_{23} = \{a_2, a_3\} \qquad \sum_{24} = \{a_1, a_3\} \qquad \sum_{34} = \{a_3, a_4\}$$

According to definition (37) we have

$$B_1(\vec{v}_1) = (\vec{p}_{11}, \vec{p}_{12}, \vec{p}_{12}, \vec{p}_{14}, \vec{p}_{11}, \vec{p}_{14}, 0, 0, 0, 0, 0, 0) \tag{45}$$

$$B_2(\vec{v}_2) = (-\vec{p}_{21}, -\vec{p}_{22}, 0, 0, 0, 0, \vec{p}_{22}, \vec{p}_{23}, \vec{p}_{21}, \vec{p}_{23}, 0, 0) \tag{46}$$

$$B_3(\vec{v}_3) = (0, 0, -\vec{p}_{32}, -\vec{p}_{34}, 0, 0, -\vec{p}_{32}, -\vec{p}_{33}, 0, 0, \vec{p}_{33}, \vec{p}_{34}) \tag{47}$$

$$B_4(\vec{v}_4) = (0, 0, 0, 0, -\vec{p}_{41}, -\vec{p}_{44}, 0, 0, -\vec{p}_{41}, -\vec{p}_{43}, -\vec{p}_{43}, -\vec{p}_{44}) \tag{48}$$

So that, indeed, the condition $\sum_{k=1}^{4} B_k(\vec{v}_k)=0$ is equivalent to

$$\vec{p}_{11} = \vec{p}_{21} = \vec{p}_{31} = \vec{p}_{41}$$

$$\vec{p}_{12} = \vec{p}_{22} = \vec{p}_{32} = \vec{p}_{42}$$

$$\vec{p}_{13} = \vec{p}_{23} = \vec{p}_{33} = \vec{p}_{43}$$

$$\vec{p}_{14} = \vec{p}_{24} = \vec{p}_{34} = \vec{p}_{44}$$

At iteration ℓ of Algorithm PD, let us call $\vec{r}^{(\ell)}$ the price vector and $\vec{v}_k^{(\ell-1)}$, k=1,2,3,4, the results of the $(\ell-1)$th iteration. The "local" problems are then:

find the local minimum $\vec{v}_k^{(\ell)}$ of criterion

$$C_k(\vec{v}_k) + \vec{r}^{(\ell)} \cdot B_k(\vec{v}_k)$$

closest to $\vec{v}_k^{(\ell-1)}$ subject to the constraint that the $\vec{p}_{ki}$'s (i=k-1,k,k+1) are probability vectors

Just as in the case of the Decomposition techniques, this example only intends to help the reader get a more concrete feeling for the concepts developed in the previous section.

Conclusions

We have presented in this paper two methods of approaching the problem of reducing the dimensionality of the Relaxation labeling problem viewed as an Optimization task. The first approach takes an algorithmic standpoint: it replaces a large dimensionality problem by a sequence of smaller dimensionality problems threaded by a sort of rubber band which forces each of them to take into account the results obtained by the others. The coordination problem is tackled directly

in the second approach in which a price is dynamically assigned to eventual discrepancies between solutions to the different "small" problems. The problem of comparing the two approaches is open but it is hoped that the rapid development of relaxation labeling applications involving large numbers of objects will provide answers to that question.

References

1. Waltz, David L., "Generating Semantic Description from Drawings of Scenes with Shadows," MIT Technical Report AI271, November 1972.

2. Rosenfeld, Azriel, Robert A. Hummel, and Steven W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-6, No. 6, June 1976, pp. 420-433.

3. Davis, Larry S. and Azriel Rosenfeld, "An Application of Relaxation Labeling to Spring-Loaded Template Matching," Proc. 3IJCPR, 1976, pp. 591-597.

4. Davis, Larry S., "Shape Matching Using Relaxation Techniques," TR 480, Computer Science Center, University of Maryland, September 1976.

5. Davis, Larry S. and Azriel Rosenfeld, "Hierarchical Relaxation for Waveform Parsing," in Computer Vision Systems, Edited by Allen R. Hanson and Edward M. Riseman, pp. 101-109, Academic Press, 1978.

6. Haralick, Robert M., "Scene Analysis, Arrangements, and Homomorphisms," in Computer Vision Systems, Edited by Allen R. Hanson and Edward M. Riseman, pp. 199-212, Academic Press, 1978.

7. Mackworth, Alan K., "Consistency in Networks of Relations," Artificial Intelligence, Vol. 8, pp. 99-118, 1977.

8. Barrow, H.G. and J.M. Tenenbaum, "Representation and Use of

Knowledge in Vision," Tech. Note 108, Artificial Intelligence Center, SRI International, Menlo Park, Ca. 1975.

9. S.W. Zucker, R.A. Hummel and A. Rosenfeld, "An Application of Relaxation Labeling to Line and Curve Enhancement," IEEE Transactions on Computers, C-26, No. 4, pp. 394-402, April 1977.

10. B. Schachter, A. Lev, S.W. Zucker and A. Rosenfeld, "An Application of Relaxation Methods to Edge Reinforcement," Computer Science Technical Report 476, MCS-72-03610, University of Maryland, August 1976.

11. A.R. Hanson, E.M. Riseman, "Segmentation of Natural Scenes," in Computer Vision Systems, pp. 129-163, Academic Press, 1978.

12. J.O. Eklundh, H. Yamamoto, A. Rosenfeld, "Relaxation Methods in Multispectral Pixel Classification," Computer Science Technical Report 662, MCS-76-23763, University of Maryland, July 1978.

13. S.T. Barnard, W.B. Thompson, "Disparity Analysis of Images," Computer Science Technical Report 79-1, Institute of Technology, University of Minnesota, January 1979.

14. D. Marr, T. Poggio and G. Palm, "Analysis of a Cooperative Stereo Algorithm," Biol. Cybernetics 28, pp. 223-239, 1977.

15. S.W. Zucker, J.L. Mohammed, "Analysis of Probabilistic Relaxation Labeling Process," Computer Vision and Graphic Laboratory, Dept. of Electrical Engineering Technical Report 78-3R, January 1978.

16. S.W. Zucker, J.L. Mohammed, "Analysis of Probabilistic Relation Labeling Processes," Proc. of the 1978 IEEE Computer Society Conference on Pattern Recognition and Image Processing, pp. 307-312, Chicago, 31 May-2 June 1978.

17. S.W. Zucker, E.V. Krishnamurthy and R.L. Haar, "Relaxation Processes for Scene Labeling: Convergence, Speed, and Stability," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-8, No. 1, pp. 41-48, January 1978.

18. O. Faugeras and M. Berthod, "Scene Labeling: An Optimization Approach," IEEE Computer Society Conference on Pattern Recognition and Image Processing, Chicago, August 1979.

19. M. Berthod and O. Faugeras, "Etiquetage par Optimisation et Application," Congres AFCET-IRIA sur la Reconnaissance des Formes et la Traitement des Images, Toulouse, September 1979.

20. O. Faugeras and M. Berthod, "Improving Consistency and Decreasing ambiguity in Stochastic Labeling: An Optimization Approach," presented to the IEEE PAMI Trans., September 1979.

21. S. Ullman, "Relaxation and Constrained Optimization by local processes," Computer Graphics and Image Processing, 10, pp. 115-125, 1979.

22. L.S. Lasdon, Optimization Methods for Image Systems, McMillan, 1970.

23. D. Wismer, Editor, Optimization Methods for Large Scale Systems with Applications, McGraw-Hill, 1971.

24. M.D. Mesarovic, D. Macho and Y. Takahara, Theory of Hierarchical Multilevel Systems, Academic Press, 1970.

25. J.L. Lions and G.I. Marchuk, Sur les Methodes Numeriques en Sciences Physiques et Economiques, Collection Methode Mathematiques de l'Informatique, Dunod, 1976.

26. P. Saint-Pierre, "Etude Theorique et Numerique de Methodes de decomposition pour des problemes elliptiques," These de 3 cycle, Faculte des Sciences d'Orsay, 1971.

2.3 Extraction of Texture Primitives

F. Vilnrotter*, R. Nevatia, K. Price

Introduction

In previous reports, we have described a program used to generate descriptions of natural textures [1-2]. (These references also contain a discussion of other work and a list of related references). The edge and direction images corresponding to the original texture image serve as inputs to the program.

In the first part of the program edge repetition arrays are produced. (An edge repetition array is in essence the binary case of a gray level co-occurrence matrix). These arrays are calculated for 6 directions (0°, 30°, 60°, 90°, 120°, and 150°), for both dark and light intensity objects, at distances within a range specified by the user. A comprehensive discussion of edge repetition arrays is given in [1].

In the second part of the program the edge repetition arrays are analyzed to determine whether there are predominant element sizes in

---

*Felicia Vilnrotter is supported by a Hughes Aircraft Company Doctoral fellowship.

any of the 6 scan directions and if so whether these elements occur at regular intervals within the image. The details of this analysis are presented in [2] and will not be repeated here.

Figure 2 is the description generated for the raffia (woven palm) texture sample given in Fig. 1. The information comprising this description is 1-dimensional in nature. The second dimension of a primitive cannot be simply obtained by examining the like intensity element sizes for other scan directions. For example, looking at Fig. 2 one might be tempted to say that raffia is made up, in part, of light elements whose dimensions are 3 pixels along its horizontal axis and 5 pixels along its vertical axis. This is, in fact, not the case. Figure 3 is an abstract representation of the basic raffia primitive cluster. The light element found in the vertical scan direction is the medium intensity block A of Fig. 3, while the light element found by the horizontal scan is the high intensity block C in Fig. 3. If both dimensions of block A were found then the vertical dimension would be characterized as belonging to a light element while the horizontal dimension would be described as belonging to a dark element. Clearly, a two dimensional description of the texture cannot be constructed by combining its one dimensional parts. However, the computed description can be used as a starting point for a two dimensional primitive search. Some other primitive extraction techniques are discussed in [3].

Primitive Extraction Process

Motivation - There are three strong indications of element size provided in the texture description given in Fig. 2. Each of these predominant element sizes is the location of a strong peak in an element size edge repetition array for the raffia image in Fig. 1. Knowing the exact locations, within the original texture image, where the edge matches contributing to these strong peaks occur would be useful. It would then be possible to isolate the uniform intensity regions or textural elements being measured. Analysis of the set of

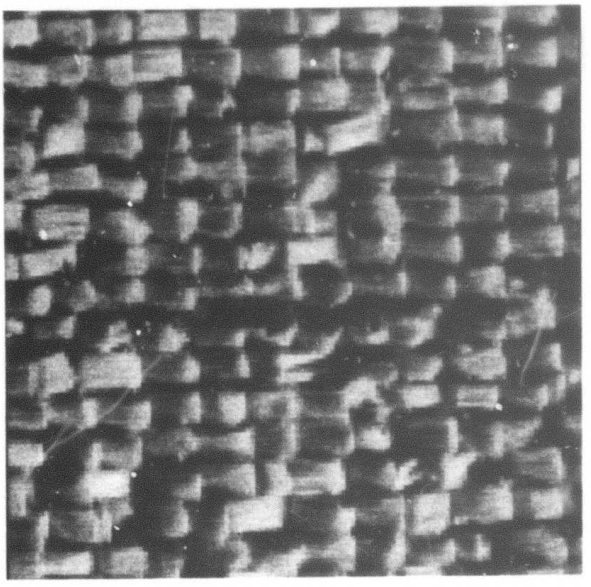


Figure 1a. Raffia Subwindow.

Figure 1b. Non-maximal suppressed edges from a.

NUMBERS APPEARING IN PARENTHESES ARE SCALE DEPENDENT

FILENAME = RAFFIA.NR10

DARK OBJECT DESCRIPTIONS

HORIZONTAL SCAN DIRECTION

NO EVIDENCE OF PERIODICITY OR PREDOMINANT ELEMENT SIZE

30 DEGREE SCAN DIRECTION

NO EVIDENCE OF PERIODICITY
WEAK EVIDENCE OF PREDOMINANT ELEMENT SIZE (25.40)

VERTICAL SCAN DIRECTION

THERE IS STRONG EVIDENCE OF PERIODICITY (ELEMENT SPACING 8.00)

THERE IS STRONG EVIDENCE OF PREDOMINANT ELEMENT SIZE (3.00)
WITH MODERATE SUPPORT FOR ELEMENT SPACING (8.00)

RATIO OF SIZE TO PERIOD IS .38

LIGHT OBJECT DESCRIPTIONS

HORIZONTAL SCAN DIRECTION

NO EVIDENCE OF PERIODICITY
STRONG EVIDENCE OF PREDOMINANT ELEMENT SIZE (3.00)

30 DEGREE SCAN DIRECTION

NO EVIDENCE OF PERIODICITY OR PREDOMINANT ELEMENT SIZE

VERTICAL SCAN DIRECTION

THERE IS STRONG EVIDENCE OF PERIODICITY (ELEMENT SPACING 8.00)

THERE IS STRONG EVIDENCE OF PREDOMINANT ELEMENT SIZE (5.00)
WITH MODERATE SUPPORT FOR ELEMENT SPACING (8.00)

RATIO OF SIZE TO PERIOD IS .63

60,120 AND 150 DEGREE SCAN DIRECTIONS FOR BOTH LIGHT AND DARK OBJECTS

NO EVIDENCE OF PERIODICITY OR PREDOMINANT ELEMENT SIZE

Figure 2. Symbolic texture Description of raffia.

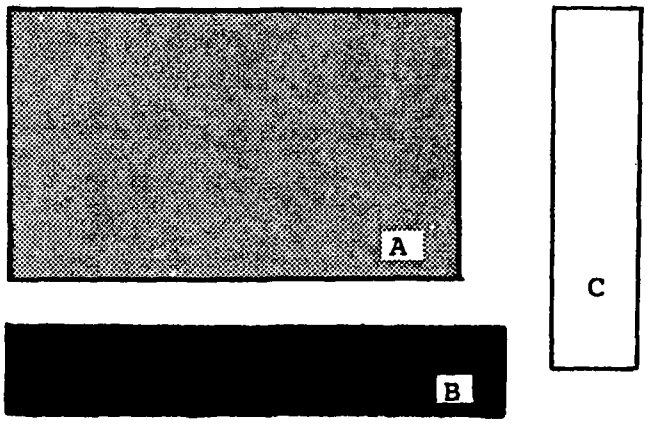


Figure 3. Abstract representations of raffia primitives.

textural elements or primitives for a particular predominant element size could provide the average intensity and area for that primitive group.

Implementation - In Fig. 2, we can see that all of the information for a texture is listed according to relative element intensity and scan direction. For example, we find that elements of size 3 and spacing 8 occur in the vertical scan direction; and these elements are dark in relation to their vertical neighbors. Since the scan direction and relative element intensity of each predominant element size is known a search for the particular set of edge matches exhibiting these restrictions can be initiated.

This search proceeds in much the same way that the original edge match search occurred. However, there are 2 exceptions:

1) This search is in one scan direction for elements with a particular relative intensity (dark or light) for a restricted range of distances.

2) Instead of recording the edge matches in an edge repetition array the 2 matching edges are marked in a blank image in locations corresponding to their locations in the original edge image. The interior points, points lying between the 2 matching edges along the direction of scan, are also marked.

In Fig. 4 the marking scheme is illustrated. Both edges are marked by "E"'s. The edge exhibiting the lower row location (and if this test fails the lower column location) is marked in a special way as a starred edge. The points between the 2 edges along the line of scan are marked as interior points, "I".

The image produced by this process is the intermediate primitive image. This image consists of zeros except where edge matches and their interior points are marked as in Fig. 4.

```
0  0  0  0  0  0  0  0
0  0  0  E* 0  0  0  0
0  0  0  I  0  0  0  0
0  0  0  I  0  0  0  0
0  0  0  I  0  0  0  0
0  0  0  E  0  0  0  0
0  0  0  0  0  0  0  0
```

Figure 4. Non-expanded primitive in vertical san direction.

```
0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0
0  1  1  1  1  1  0  0
0  0  1  1  1  1  1  0
0  1  1  1  1  1  0  0
0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0
```

Figure 5. Expanded primitive for Figure 4.

The next step is the expansion of the interiors of the primitive slices to form a binary mask image for the primitive set. In the expansion process the original texture image as well as the edge and direction images are used.

The intermediate primitive image is searched until an "E" is encountered. Proceeding in the direction of scan the first interior point, "I", is encountered. A "1" is placed in the binary mask image at this point and the texture image intensity of this point is noted. This is repeated until the edge mark at the end of the primitive slice is encountered. The range of intensity values for the interior pixels of the primitive slice is now known. At this point the primitive interior can be expanded to its "natural boundary." The natural boundary of a primitive is made up of:

1) The edges within the edge image;

2) pixels whose intensities are outside of the interior intensity range by more than 10; and,

3) the boundary of the texture image itself.

Expansion of the primitive slice proceeds outward from the interior points of the primitive slice in a direction perpendicular to the direction of scan. Figure 5 shows the expanded primitive within the primitive mask image corresponding to the primitive slice in Fig. 4.

The expanded primitives within the mask image can then be analyzed to determine an average primitive intensity, an average primitive area in pixels and from this area the average primitive dimension in the direction perpendicular to the line of scan.

Results

Results generated for the raffia sample of Fig. 1 are given in Fig. 6. The dark primitive found in the vertical scan direction corresponds to block B in Fig. 3. The light primitive found in the vertical scan direction corresponds to block A in Fig. 3 and the light primitive found by the horizontal scan corresponds to block C. The first primitive dimension given in each of the primitive descriptions is the dimension along the line of scan, while the second dimension is in a direction perpendicular to the scan line. Figures 7 through 9 are the primitive mask images corresponding to the primitives of type A, B, and C of Fig. 3, respectively. Figure 10 is composed of all 3 types of primitives with their average intensity values kept constant.

Work on this program is still in progress. It is hoped that information obtained in this way will lead to the generation of more meaningful texture descriptions and will be useful in the texture classification process.

References

1. R. Nevatia, K. Price and F. Vilnrotter, "Describing Natural Textures," USCIPI Semiannual Technical Report #860, March 1979, pp. 29-54.

2. F. Vilnrotter, R. Nevatia and K. Price, "Automatic Generation of Natural Texture Descriptions," USCIPI Semiannual Technical Report 910 September 1979, pp. 31-63.

3. A. Rosenfeld, "Cooperative Computation in Texture Analysis," in Proc. Image Understanding Workshop, Los Angeles, Nov. 1979, pp. 52-56.

```
PRIMITIVE ANALYSIS FOR TEXT. SUPP12  (THRESH = 10)


RELATIVE INTENSITY IS DARK      DIRECTION IS VERTICAL

NUMBER OF SAMPLES: 108

  AVERAGE PRIMITIVE DIMENSIONS ARE: (2.00 AND 10.39)

  AVERAGE PRIMITIVE SIZE IN PIXELS IS: (20.30)

  AVERAGE PRIMITIVE INTENSITY IS: (128.79)

  PRIMITIVES REPEAT AT ELEMENT SPACING: (8.00) IN ABOVE MENTIONED DIRECTION


RELATIVE INTENSITY IS LIGHT     DIRECTION IS VERTICAL

NUMBER OF SAMPLES: 109

  AVERAGE PRIMITIVE DIMENSIONS ARE: (4.00 AND 9.33)

  AVERAGE PRIMITIVE SIZE IN PIXELS IS: (36.94)

  AVERAGE PRIMITIVE INTENSITY IS: (172.35)

  PRIMITIVES REPEAT AT ELEMENT SPACING: (8.00) IN ABOVE MENTIONED DIRECTION


RELATIVE INTENSITY IS LIGHT     DIRECTION IS HORIZONTAL

NUMBER OF SAMPLES: 68

  AVERAGE PRIMITIVE DIMENSIONS ARE: (2.00 AND 7.88)

  AVERAGE PRIMITIVE SIZE IN PIXELS IS: (15.18)

  AVERAGE PRIMITIVE INTENSITY IS: (190.47)

  NO EVIDENCE OF PERIODICITY
```

Figure 6. Raffia primitive texture element description.

Figure 7. Light raffia primitive found in vertical scan.

Figure 8. Dark raffia primitive found in vertical scan.

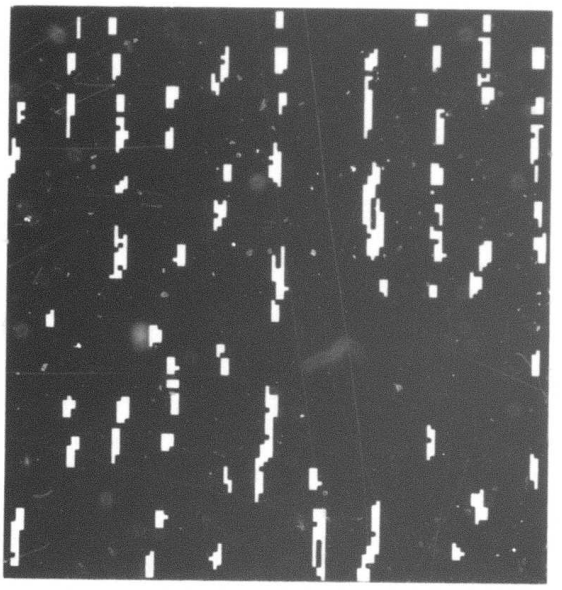


Figure 9. Light raffia primitive found in horizontal scan.

Figure 10. Composite primitives.

2.4 Texture Edge Detection

O.D. Faugeras and H.Y. Lee

Background

The problem of detecting texture edges is brought up in many tasks of image analysis where objects are present that differ mostly in texture. Conventional edge detection techniques based on detection of discontinuities in intensity usually fail to locate boundaries between such objects. Thus, the need for developing new techniques that will perform better in such cases comes out.

Obviously, such techniques should be based on some sort of texture measurements. The quality that differentiates texture properties from intensity properties is that the first ones are local that is measured over some neighborhood of every pixel whereas the second ones are pixel properties.

This is the first problem encountered when working with texture. The second problem comes in as soon as we ask ourselves the question of how to measure texture properties. To make a long story short, let us say briefly that there are two main approaches toward texture analysis. The deterministic approach considers textures as a periodic or pseudoperiodic repetition of some base pattern whereas the stochastic approach prefers to view texture fields as samples of stochastic processes. As a consequence we can find in the literature two broad classes of techniques corresponding to these two approaches which make use of different tools: spectral analysis, syntactic methods in the deterministic case, statistical methods in the

stochastic case. For a recent review of the different approaches toward texture analysis see [1].

Based on previous work by Julesz [2] and Pratt et al. [3], Faugeras and Pratt [4] proposed a technique of texture feature extraction based on measurements of the autocorrelation function of the texture field and the first order histogram of the decorrelated field. Another technique was proposed by Faugeras [5] based on a human visual model where textures are analyzed through a bank of nonlinear channels. Laws [6] expanded on both works and proposed to analyze textures with a set of filters of small spatial extent (analogous to the decorrelation operator in [4] or the bandpass filters in [5]) and compute local "energy" values in the output planes after convolution (analogous t the L -norms in [5]). If N filters are used in the process, texture is represented at every pixel by an N-dimensional vector. Moreover, Laws has been able to exhibit a limited set of filters (of the order of 4) that seem to perform best in terms of classification accuracy on a fairly large set of natural texture. Because of its simplicity and good performance, we adopted Laws' method of texture feature extraction summarized in Fig. 1.

The problem of detecting texture edges is therefore equivalent to the problem of detecting edges in the "energy" images $I_n(k,\ell)$ for $n=1,\ldots,N$. Edge detection methods fall in two broad classes. Techniques based on enhancement of edges followed by thresholding are probably the simplest ones while techniques based on local filtering of an edge model are slightly more complicated. For a good review of existing techniques, see Davis [9] and for a quantitative evaluation of many methods falling in the first class, see Abdou and Pratt [7]. Two complications arise when working with textures rather than intensities, one of them we already mentioned before it is linked to the fact that texture measures are local, the other one stepping from the fact that we have to deal with vector-valued functions rather than scalar-valued functions very much like for the detection of color edges. In other words, we should be able to exploit the

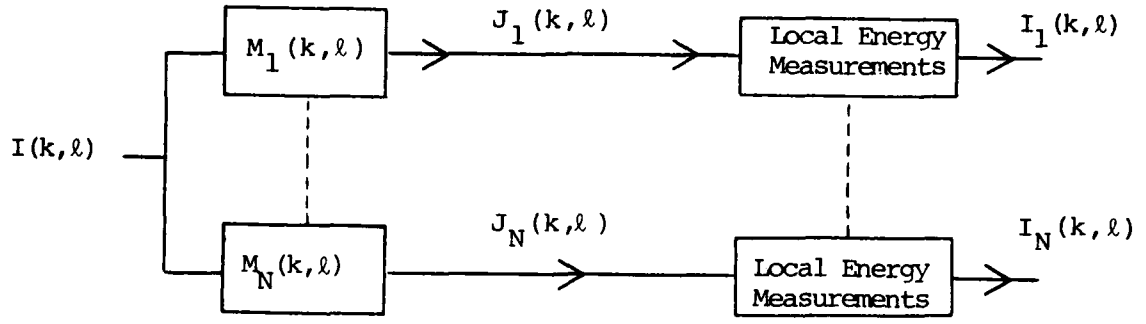


Figure 1. Texture feature extraction by parallel processing of an image with convolution masks $M_1(k,\ell),\ldots,M_N(k,\ell)$. The local energy measurements are given by

$$I_n(k,\ell) = \sum_{i,j} |w_n(k-i,\ell-j)J_n(i,j)|^P \qquad n=1,\ldots,N$$

where P is a positive integer and the window functions w_n may or may not depend on n.

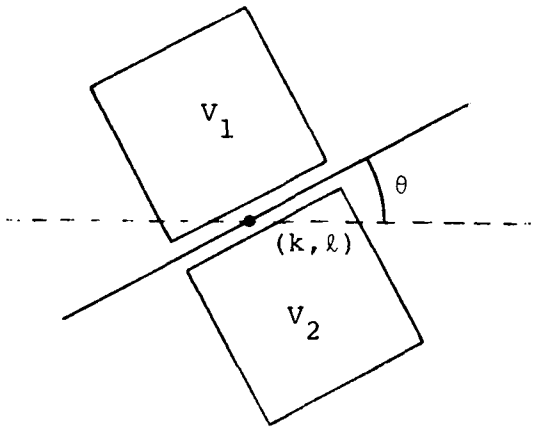


Figure 2. Energy measures are computed in neighborhoods V_1 and V_2 of pixel (k,ℓ). The magnitude of their difference is an indication of the presence/absence of a texture edge in the direction θ at that pixel.

cross-correlation between the image planes $I_n(k,\ell)$ to improve our edge detection algorithm.

Texture Gradient Computation

The method we propose for detecting texture edges if patterned after the methods for detecting intensity edges using directional derivatives. We can think of this process as acting either on the texture energy planes $I_n(k,\ell)$ of Fig. 1 or on the output of the convolution masks $J_n(k,\ell)$. The idea, similar to what has been described in [8,9] is that if we wish to detect a texture edge in the direction θ (modulo π) at pixel (k,ℓ), we would want to subtract energy measured computed in neighborhoods V_1 and V_2 of that pixel as described in Fig. 2. This is repeated for different values of θ, and thus providing an indication of the presence/absence of a texture edge at every pixel. The actual detection is performed by thresholding or local maximum selection.

Now since we have several texture energy planes to work with, the question arises naturally of how to combine the edge information coming from those different sources. A simple way of approaching this problem would be to select the "best" frames in terms of some criterion based on smoothness of edges and interimage edge element correlation. Another potential candidate is to use some sort of relaxation labeling process [10,11] that would combine evidence from the different texture planes in a meaningful fashion.

Preliminary Results

We have applied some of the ideas to the composite image showed in Fig. 3. It is made of two natural textures, sand and water, whose histograms have been normalized. This picture has been processed by the four 5x5 masks described in Fig. 4. Vertical and horizontal texture edges have been detected in all four planes and results are shown in Fig. 5. Obviously, masks #1 and #2 are performing much

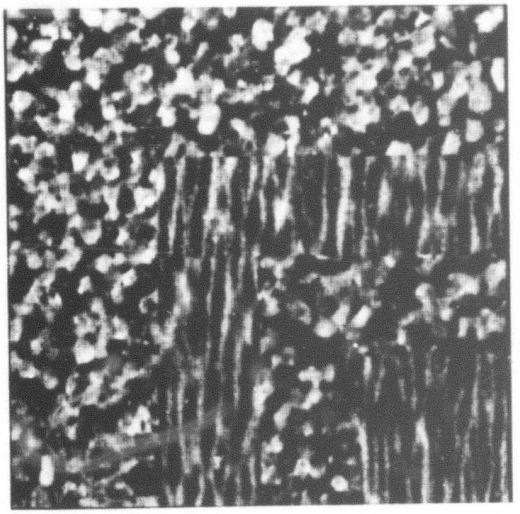


Figure 3. Composite image of water and sand.

-1	-4	-6	-4	-1
-2	-8	-12	-8	-2
0	0	0	0	0
2	8	12	8	2
1	4	6	4	1

Mask #1

-1	0	2	0	-1
-2	0	4	0	-2
0	0	0	0	0
2	0	-4	0	2
1	0	-2	0	1

Mask #2

1	-4	6	-4	1
-4	16	-24	16	-4
6	-24	36	-24	6
-4	16	-24	16	-4
1	-4	6	-4	1

Mask #3

-1	0	2	0	-1
-4	0	8	0	-4
-6	0	12	0	-6
-4	0	8	0	-4
-1	0	2	0	-1

Mask #4

Figure 4. Texture feature extraction masks.

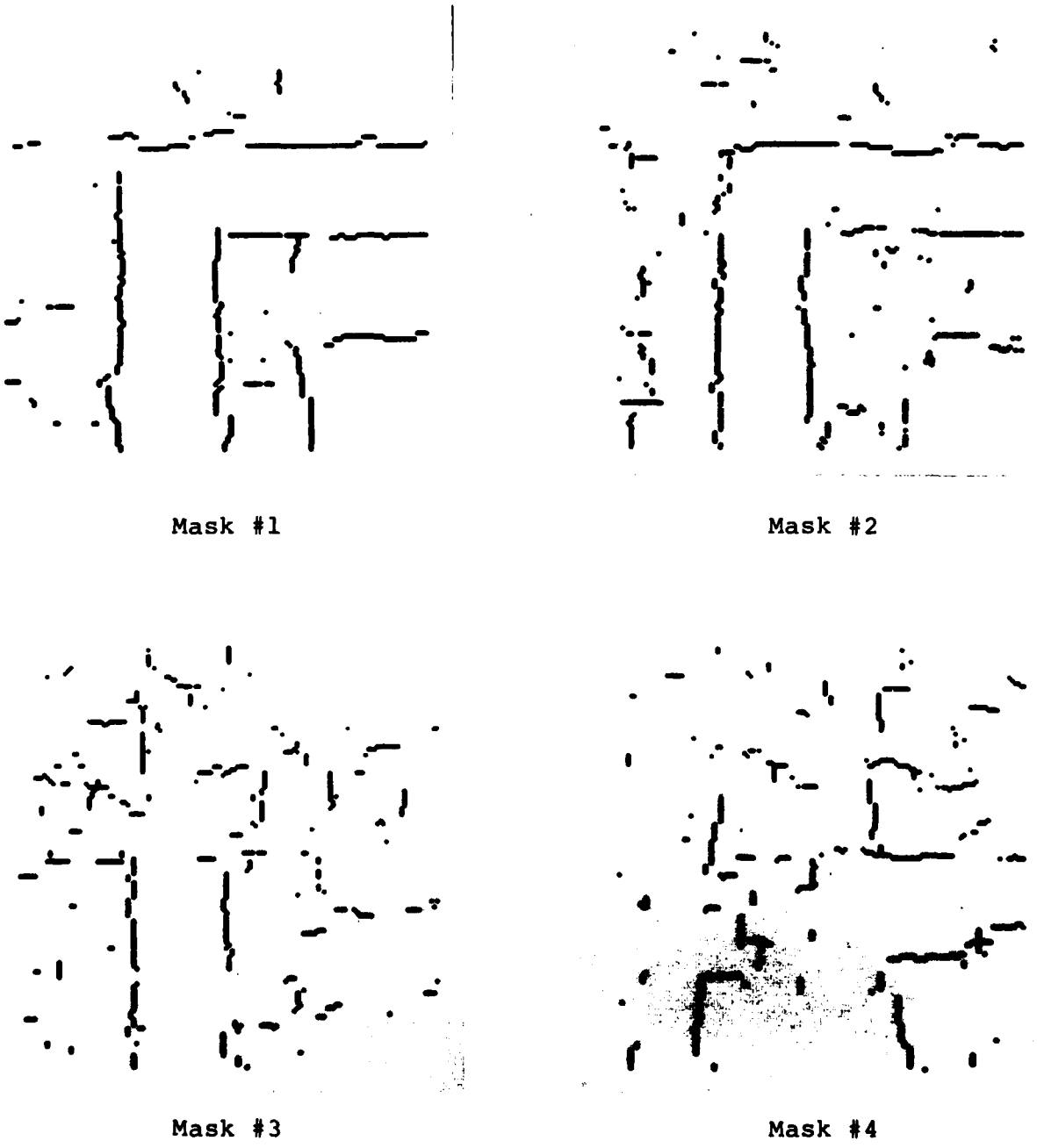


Figure 5. Results of the texture edge detection algorithm.

Figure 6. Results obtained by combining information obtained from Mask #1 and Mask #2.

better than the other two masks and Fig. 6 shows the improvement obtained in combining the results of the first two masks.

As a conclusion, we believe that texture edges can be accurately extracted by methods of the type of the one described in this report. We are concentrating now on developing efficient techniques of combining the information extracted by the different masks and the quantitative analysis of the performances of a texture edge detection algorithm based on these ideas.

References

1. R.M. Haralick, "Statistical and Structural Approaches to Texture," Proceedings of the IEEE, pp. 786-804, May 1979.

2. B. Julesz, "Visual Pattern Discrimination," IRE Transactions on Information Theory, Vol. IT-8, No. 1, pp. 84-92, February 1962.

3. W.K. Pratt, O.D. Faugeras and A. Gagalowicz, "Visual Discrimination of Stochastic Texture Fields," IEEE Transactions on Systems, Man, and Cybernetics, November 1978.

4. O.D. Faugeras and W.K. Pratt, "Decorrelation Methods of Texture Feature Extraction," to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence, 1980.

5. O.D. Faugeras, "Texture Analysis and Classification Using a Human Visual Model," IJCPR Proceedings, pp. 549-559, Kyoto, Japan, November 1978.

6. K.I. Laws, "Textured Image Segmentation," Department of Electrical Engineering, Ph.D. Dissertation, January 1980.

7. I.E. Abdou and W.K. Pratt, "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors," Proceedings of the IEEE,

pp. 753-763, May 1979.

8. M. Yachida, M. Ikeda and S. Tsuji, "Boundary Detection of Texture Regions," Proc. of the 6th international Joint Conference on Artificial Intelligence, pp. 992-994, 1979.

9. L.S. Davis and A. Mitiche, "Edge Detection in Textures," Computer Graphics and Image Processing, Vol. 12, No. 1, pp. 25-39, January 1980.

10. A. Rosenfeld, R.A. Hummel and A.S. Zucker, "Scene Labeling by Relaxation Operations," IEEE Transactions on Systems, Man, and Cybernetic, Vol. SMC-6, No. 6, June 1976, pp. 420-433.

11. O.D. Faugeras and M. Berthod, "Improving Consistency and Decreasing Ambiguity in Stochastic Labeling: An Optimization Approach," submitted to the IEEE Transactions on Pattern Analysis and Machine Intelligence, September 1979.

2.5 Application of the General Linear Model to Binary Texture Synthesis

D.D. Garber

Introduction

Texture is an important characteristic for the analysis of images. Accordingly, the study of this image feature has been extensive and this has led to the development of models which attempt to explain and describe this quality of an image field.

Earlier results [1] have indicated the usefulness of both Markov and linear autoregressive models in attempting to describe and simulate natural textures. In the Markov model, a set of generation parameters was used to generate a two-dimensional array of binary pixels line by line using a set of generation parameters

$$G_{V_{N+1}}(V_1,V_2,\ldots,V_N) = P(V_{N+1}|V_1,V_2,\ldots,V_N) \qquad (1)$$

where $P(A|B)$ represents the probability of A given B. Each pixel value, V_{N+1}, depends on the N pixels previous to it. By allowing these previous pixels, V_i, to be on not only the same image line as V_{N+1} but also on lines above it (Figure 1), horizontal as well as vertical dependence is induced into the generating process. This method of texture generation soon finds its limitations as 2^{N+1} generation parameters must be accounted for in a binary case and g^{N+1} in a g grey-level case. It was found that for a fixed value of N, better texture simulations could be obtained by letting the pattern of the V_i's to become flexible and noncontiguous (Figure 2).

This idea was also used in the linear-model method of texture synthesis. The linear model may be expressed in normal linear regression form as

$$Y_i = \vec{X}_i\vec{\beta}+\varepsilon_i \qquad i = 1,2,\ldots,n \qquad (2)$$

where

$$Y_i = V_{N+1,i} \qquad \vec{X}_i^T = \begin{bmatrix} V_{1,i} \\ V_{2,i} \\ \cdot \\ \cdot \\ \cdot \\ V_{N,i} \\ 1 \end{bmatrix} \qquad X = \begin{bmatrix} X_1 \\ X_2 \\ \cdot \\ \cdot \\ \cdot \\ X_N \end{bmatrix}$$

$\vec{\beta}$ is an (N+1)x1 vector of unobservable parameters and ε_i is an unobservable random variable such that $E[\varepsilon_i]=0$, The most common

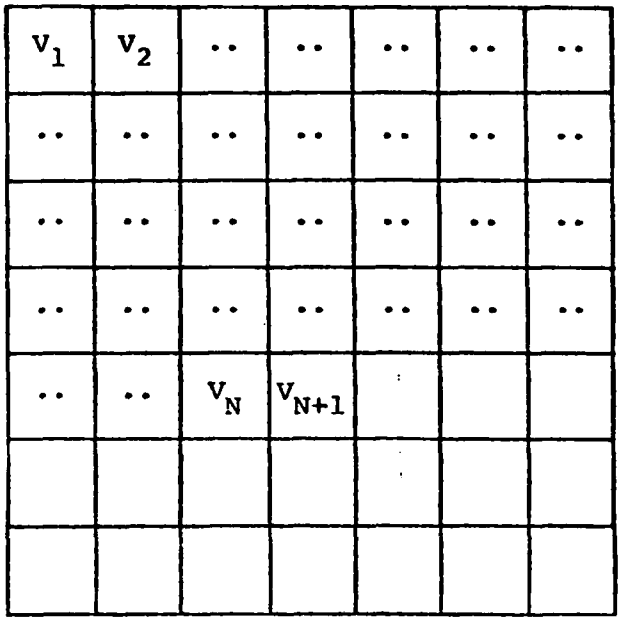


Figure 1. Two-dimensional synthesis.

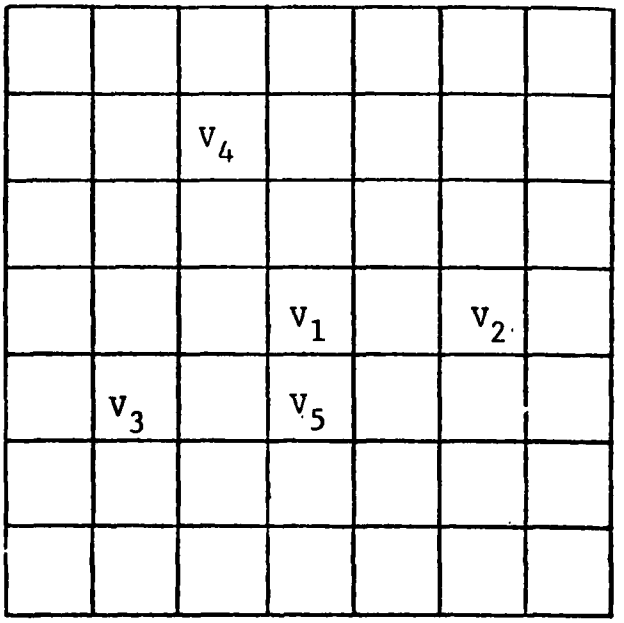


Figure 2. One-dimensional synthesis.

estimator of β is $(X'X)^{-1}X'Y$, the least-squares estimator. So given a parent texture which we desire to simulate, an estimation of the texture model parameter β, $\hat{\beta}$, may be obtained yielding the generation model

$$\hat{Y} = X\hat{\beta} \tag{3}$$

Noise is added to our model during the generation process which is similar to the noise we measure in the original parent texture. It is very important to note that the estimation of the amount of noise present in a texture is usually obtained by measuring the amount of variation in the parent texture which is unexplained by our model. Therefore, as in any modeling process, any shortcomings or inaccuracies of the model will appear to be "noise" (unexplained variance) and hence the amount of noise will increase. In [1] uniform gaussian noise was used to drive the linear system and the variance of this noise was estimated by measuring the variation of the parent texture which was unexplained by the linear model when that linear model was applied to it.

Choosing the V_i's to be in the kernel

We will refer to the V_i's on which the next pixel, V_{N+1}, depend as the kernel of the generation process. Geometrically speaking, the V_i's form a kernel "shape" which may or may not be spatially contiguous. For example, in Figure 2 a generating kernel shape is shown where the V_5 pixel depends on only some of the pixels in its surrounding neighborhood. In this case, V_5 may be generated based on the values of pixels V_1, V_2, V_3 and V_4 but is dependent on no other pixels in the neighborhood. This does not imply that V_5 is not related or correlated with its other neighbors. In fact, the relationships between V_1, V_2, V_3, V_4 and V_5 will determine other interrelationships.

A non-contiguous neighborhood of V_i's is used as it allows a more parsimonious model for texture generation to be chosen. An analogy in the simple linear regression case would be the dropping of independent variables which do not contribute to the prediction or estimation of the dependent variable. In texture generation this allows the model to be estimated by fewer parameters and makes the generation-synthesis process more efficient. It also relieves us from the complex numerical problems of analyzing and inverting matrices of unwieldy size, a necessary step in linear model parameter estimation. We would, for example, not expect our V_{N+1} pixel to depend on a pixel V_i where the spatial separation between V_{N+1} and V_i is large. If that distance is small, however, we would expect a large dependence.

Methods for choosing the proper independent variables (V_i's) to be included in the generation process have been developed over many years [3,4,5] for other applications. If we limit ourselves to a finite neighborhood of M variables we would wish to choose the best subset of N variables, where N<M and is more manageable in size. Evaluating such subsets and their corresponding models requires a criterion function. The most commonly used is mean-square error. The problem then is to choose a subset of V_i's of size N from a set of V_i's of size M (M>N) such that the model $Y=X\beta+\varepsilon$ employing those V_i's produces the minimum mean-square error when compared to all other possible models containing N V_i's. The only possible way to do this is to evaluate all possible $\binom{N}{M}$ combinations of points. Even for small values of M and N this would require enormous computation power. For example, for M=40 and N=12, over 5.5 billion models would have to be evaluated! Therefore a sub-optimal approach which yields a good solution but not necessarily the best solution, must be used.

Correlation and Partial Correlation

We may define the mean and variance of a variable X to be

$$\mu_X = E[X] \tag{4}$$

and

$$\sigma_X^2 = E[(X-\mu)^2] \tag{5}$$

Similarly we may define the covariance of two variables X_1 and X_2 as

$$\begin{aligned} \gamma_{X_1X_2} &= E[(X_1-\mu_{X_1})(X_2-\mu_{X_2})] \\ &= E[(X_1X_2)-\mu_{X_1}\mu_{X_2}] \end{aligned} \tag{6}$$

And their correlation coefficient as

$$\rho_{X_1X_2} = \frac{\gamma_{X_1X_2}}{\sigma_{X_1}\sigma_{X_2}} \tag{7}$$

Using the above definitions we may then define the partial correlation coefficient of variables X_1 and X_2 after both have been adjusted for X_3 as

$$\rho_{12\cdot3} = \frac{\rho_{12}-(\rho_{13})(\rho_{23})}{\sqrt{1-\rho_{13}^2}\sqrt{1-\rho_{23}^2}} \tag{8}$$

Each of the above parameters has a corresponding statistic (estimate of a parameter of a population given an observable sample of that population) which is chosen to meet some desirable set of a criteria such as sufficiency, consistency and unbias of estimate under certain conditions. These would be

$$\hat{\mu}_X = \frac{\Sigma X}{N} = \bar{X} \tag{9}$$

$$s^2 = \frac{\Sigma (X_i - \bar{X})^2}{N-1} \tag{10}$$

$$r_{X_1X_2} = \frac{\sum_i (X_{1i} - \bar{X}_1)(X_{2i} - \bar{X}_2)}{\{[\sum_i (X_{1i} - \bar{X}_1)^2][\sum_i (X_{2i} - \bar{X}_2)^2]\}^{1/2}} \tag{11}$$

$$r_{X_1X_2 \cdot X_3} = \frac{r_{X_1X_2} - (r_{X_1X_3})(r_{X_2X_3})}{\sqrt{1-r^2_{X_1X_3}}\sqrt{1-r^2_{X_2X_3}}} \tag{12}$$

Second-order partial autocorrelation may be found in a similar manner using

$$r_{X_1X_2 \cdot X_3X_4} = \frac{r_{X_1X_2 \cdot X_3} - r_{X_1X_4 \cdot X_3} r_{X_2X_4 \cdot X_3}}{\sqrt{1-r^2_{X_1X_4 \cdot X_3}}\sqrt{1-r^2_{X_2X_4 \cdot X_3}}} \tag{13}$$

Higher order partial correlations maybe found by extension of the above [7].

The Forward Selection Procedure

One approach to finding a good subset of independent variables is known as the forward selection procedure. This method inserts N variables into the model one-at-a-time. The order of insertion is determined by using the partial-correlation coefficient as a measure of the importance of variables not yet in the equation. The basic procedure is as follows. First we select the V_{i_1} most correlated with V_{N+1} and we find the first-order linear regression equation $\hat{V}_{N+1}=aV_{i_1}+b$. We next find the partial correlation coefficient of V_j (for all $j=i$) and V_{N+1} (after allowance for V_i). Mathematically this is equivalent to finding the correlation between the residuals from the regression $\hat{V}_{N+1}=a_1V_{i_1}+a_2$ and the residuals from another regression $V_i=b_1V_{i_1}+b_2$ (which we have not actually performed). The V_j with the highest partial correlation coefficient with V_{N+1}, V_{i_2} is now selected and a second equation $\hat{V}_{N+1}=c_1V_{i_1}+c_2V_{i_2}+c_3$ is fitted to the data. This process continues. After $V_{i_1},V_{i_2},\ldots,V_{i_q}$ are in the regression the partial correlation coefficients are the correlations between the residuals from the regression $\hat{V}_{N+1}=f(V_{i_1},V_{i_2},\ldots,V_{i_q})$ and the residuals from a regression $\hat{V}_{i_j}=f_j(V_{i_1},V_{i_2},\ldots,V_{i_q})$ $(j>q)$. The V_{i_j} with the highest partial correlation coefficient is now selected for entry into the linear model. The process is continued until N V_i's are entered into the model.

The final N variables chosen by a forward selection procedure is not guaranteed to be the optimal set but given the logistics of the selection procedure, the solution obtained is usually close to optimal.

Numerical Implementation

One of the most common procedures for implementing the forward selection process numerically utilizes Doolittle decomposition [5]. The Doolittle decomposition may be used to find the inverse of the correlation matrix, R, and the estimates of linear model coefficients as each variable is entered in the model. The correlation matrix is factored into the product of two triangular matrices

$$R_P = L_P U_P \tag{14}$$

where L_P is lower triangular and U_P is upper triangular with ones on the diagonal. Partial autocorrelation coefficients may be obtained easily from elements of matrix during its decomposition at each step. For further details and examples see Beyer [4] and Draper and Smith [3].

Choosing N

In the practical case N is often chosen by computational limits imposed by finite processing capability or finite memory. The ideal of parsimony would also urge the selection of the smallest N possible. In the most general texture synthesis (Markov) model which utilizes N-grams [1] we find that as many as g^N storage locations are required in order to collect the data needed to synthesize a texture from a parent where g is the number of grey levels in the texture. This approach calls for N to be less than 17 for g=2 (a binary image) if we have only 2^{17}=131,072 storage locations in memory. Approaches to "stretch" this limitation have been investigated and will be discussed in later papers. If we have 8 grey levels then N must be less than or equal to 5 as 8^5=32,768. Thus, in synthesizing textures using an N-gram approach, processor storage capability is the major limiting factor.

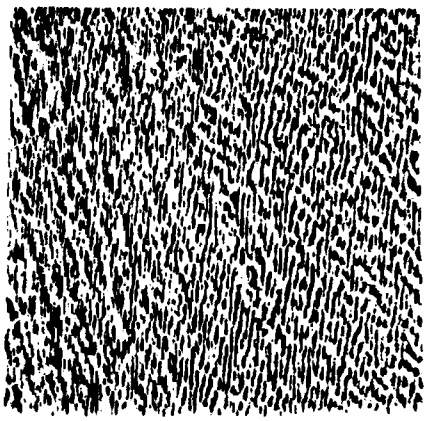
The general linear model approach to synthesizing textures was developed to overcome this limitation and produce a simpler model. However, as was discussed earlier, model parameter estimation requires the inversion of an $(N+1)^2$ matrix where N is the number of variables in the model (1).

Application to Binary Textures

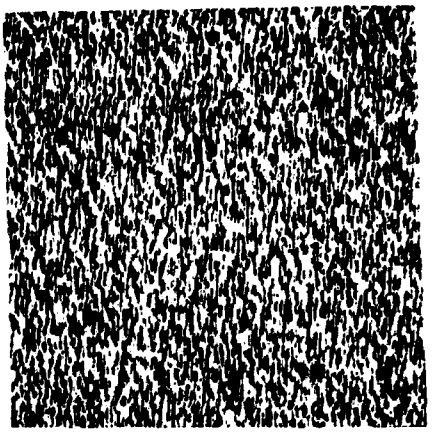
The linear autoregressive approach may be used to generate binary textures as well as continuous-tone textures. Applying such a model to binary textures corresponds closely to using a linear discriminant function in a two-class discrimination problem. A decision (white or black) is made based on a linear combination of previous measurements. Two parent textures water (Figure 3a) and raffia (Figure 4a) were chosen and estimates of parameters of the linear models were determined. Figure 3b (N=56) and 4b (N=70) show the binary texture generated using the resulting linear model. The noise used in the generation process was normally distributed, although better choices in this case could possibly be found. At each pixel generation, the value of the next pixel (obtained by taking a linear combination of previous binary pixels in its neighborhood (Figure 2) plus noise) was quantized (thresholded) to a binary value (white or black) before generating the next pixel. Texture synthesis using the more complex Markov model are also shown in figures 3c and 4c. Additional methods of synthesizing binary textures using the ideas of Cox [8] are currently being investigated.

Application to Discrimination and Identification Textures

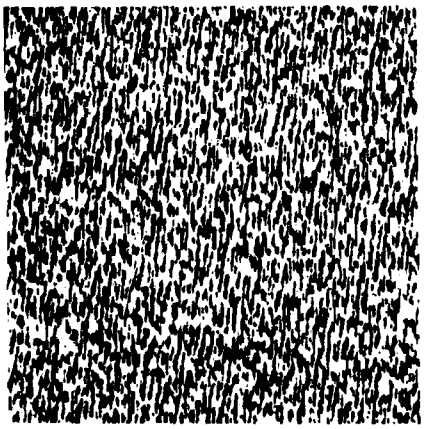
By making assumptions concerning the distribution of the error term, $\vec{\varepsilon}$, in the linear model $Y=\beta X+\varepsilon$ which is fit to the data, probability distributions may be derived for the statistics of the regression (autoregressive) model. It has been shown by Graybill [2] and Box and Jenkins [6] that the statistical estimate of β, $\hat{\beta}$ has mean β and covariance



(a) Original water.

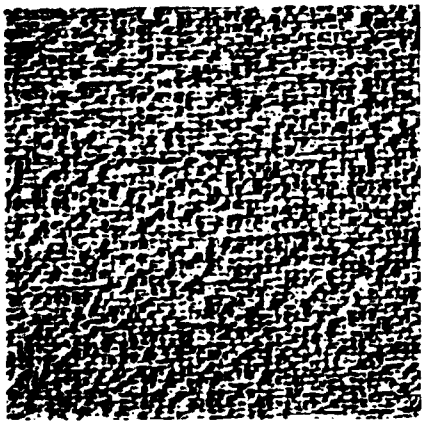


(b) Linear-model water.

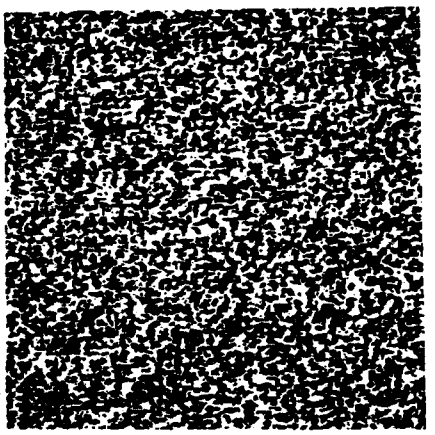


(c) Markov water.

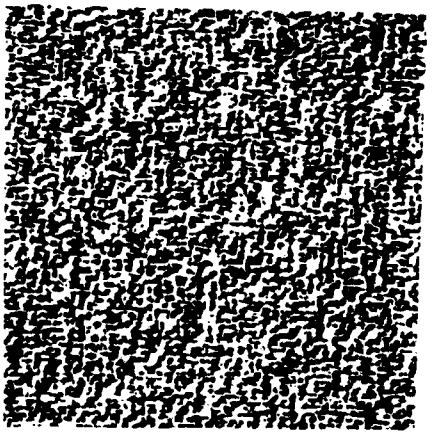
Figure 3.



(a) Original raffia.



(b) Linear-model raffia.



(c) Markov raffia.

Figure 4.

$$\text{cov}[\hat{\beta}] = n^{-1}\sigma_\varepsilon^2\Gamma^{-1} \tag{15}$$

where Γ is the auto covariance matrix of the N successive values of the time series process. To derive (15) it is necessary to assume that the elements of $\vec{\varepsilon}$ are independent with constant variance σ_ε^2. Estimates of the variances and covariances are obtained by substituting estimates for the parameters in (15) yielding

$$\hat{\text{cov}}[\hat{\beta}] = n^{-1}\hat{\sigma}_\varepsilon^2\hat{\Gamma}^{-1} \tag{16}$$

$$\hat{\Gamma} = n^{-1}(X^T X)$$

where

$$X = \begin{bmatrix} z_{-N+1} & z_{-N+2} & z_{-N+3} \cdots & z_0 \\ z_{-N+2} & z_{-N+3} & z_{-N+4} \cdots & z_1 \\ \vdots & & & \vdots \\ z_{n-N} & & \cdots & z_{n-1} \end{bmatrix} \tag{17}$$

and

$$\hat{\sigma}^2 = n^{-1}\left\{\sum_{t=1}^{n}\left[z_t - n^{-1}\sum_{u=1}^{n} z_u\right]^2\right\}$$

For a two-dimensional time series such as a texture, the z_i are obtained by passing the chosen generation kernel containing N points over the texture of interest n and measuring the values, V_i, at each of the the n positions.

Given this fact, it is easy to test various hypotheses about β and confidence limits may be obtained on its individual elements if a distribution of the errors ε_i is assumed. Sub-model tests incorporating subsets of the β vector may also be tested. Basically, at each step of the forward selection procedure, a hypothesis testing whether $\beta_i=0$ is performed where $\hat{\beta}_i$ is the estimated linear coefficient corresponding to the newly-entered variable V at that step. This is accomplished by observing the test statistic

$$F = \frac{\text{incremental sum of squares due to } V_i/1}{\text{sum of squares residual}/(N-i-1)} \tag{18}$$

This statistics has an F distribution with degrees of freedom 1 and (N-i-1) when the errors, ε_i, of the model are normally and independently distributed. When all V_j's not included in the model at step i fail to meet the test of significance the process of adding variables to the model may be stopped. Unfortunately, halting the process at this point does not insure statistical of all variables entered to that point nor does it insure non-significance of variables not entered. This problem, as that of finding the optimal set of N variables, may be solved only by considering all possible models.

The distribution of the β_i's can be used to test the equivalence of two linear models and therefore, the equivalence of the textures which those models attempt to explain. Application of this concept to identification and discrimination of texture fields will be done in future work.

Conclusion

The application of the concept described in this paper led to a good simulation of a variety of textures using both the Markov and linear autoregressive model [1]. An approach for applying the linear model to the problem of texture identification and discrimination was

proposed and is currently being investigated.

References

1. D.D. Garber, "Models for Texture Analysis and Synthesis," Technical Report USCIPI 910, 1979.

2. F.A. Graybill, Theory and Application of the Linear Model, Duxbury Press, Massachusetts, 1976.

3. N. Draper and H. Smith, Applied Regression Analysis, Wiley, New York, 1966.

4. W.H. Beyer, Handbook of tables for Probability and Statistics, The Chemical Rubber Company, Cleveland, 1974.

5. G.W. Stewart, Introduction to Matrix Computations, Academic Press, New York, 1973.

6. G.E.P. Box and G.M. Jenkins, Time Series Analysis: Forecasting and Control, Holden-Day, San Francisco, 1976.

7. D.F. Morrison, Multivariate Statistical Methods, McGraw-Hill, New York, 1967.

8. D.R. Cox, The Analysis of Binary Data, Chapman and Hall, London, 1970.

2.6 Algebraic Reconstruction Techniques for Texture Synthesis

O.D. Faugeras and D.D. Garber

Introduction

Julesz's conjecture [1] that second-order statistics are sufficient in terms of human visual texture discrimination provides a useful bound on the amount of information necessary to reconstruct a texture field. Although counterexamples to that conjecture have recently been found [2,3], it still seems to be a fairly accurate first-order approximation. Examples of the use of that upper bound for texture analysis can be found in [4,5]. Therefore it is very tempting to use it for synthesizing natural texture fields. Previous approaches to texture synthesis have used the nth-order Markov model approach [3,6,7], the autoregressive model approach [5,8,9] or the syntactic approach [10].

The purpose of this paper is to show that if we use the generation coefficient approach (to be defined below) to synthesize textures, even though we limit our knowledge of the original random field to second-order statistics (and in fact to kth-order statistics, $k \geq 2$) we are confronted to the task of "inventing" higher-order statistics. We will demonstrate how this can be done using Algebraic Reconstruction Techniques (ART).

Problem Definition

The stochastic approach toward Texture Analysis considers texture fields as samples of 2-D stochastic fields. Assuming that we are dealing with sampled and quantized imagery, let $f(n_1,n_2)$ denote the random field. n_1 and n_2 are integers representing coordinates of points in the plane. Let $\vec{n}$ be the vector of coordinates n_1 and n_2. Second-order statistics are given by the set of second-order joint

density functions

$$p_{\vec{n},\vec{m}}(x_1,x_2) \tag{1}$$

for all possible vectors $\vec{n}$ and $\vec{m}$, where x_1 and x_2 are the values of the random variables $f(\vec{n})$ and $f(\vec{m})$, respectively. We will also assume for simplicity that the random field is homogeneous, that is, invariant through translations:

$$p_{\vec{n},\vec{m}} \equiv p_{\vec{n}-\vec{m}} \tag{2}$$

Suppose that we are given the set of joint density functions $p_{\vec{r}}$ for all vectors $\vec{r}$ and that we want to synthesize a texture field with the same second-order statistics. Since we have to work with finite images, we will assume that $1 \leq n_1, n_2 \leq N$.

There are many ways of carrying out such a synthesis. If we stick to a television type of scanning (left to right, top to bottom), we will first generate f(1,1) using first-order statistics, then f(1,2) using second-order statistics, f(1,3) using third-order, etc... Even if we assume finite memory, namely that intensity at joint (i,j) depends only on intensities at points located in some finite neighborhood, we see that second-order statistics are not sufficient to generate the process. Indeed, assuming a memory of order P, the intensity at joint (i,j) is computed using the conditional probability or generation coefficient

$$p(x_{ij}|x_{i_1j_1},\ldots,x_{i_Pj_P}) \tag{3}$$

which involves (P+1)th-order joint density functions. Therefore if P is larger than 1, we have to "invent" those higher-order densities.

Another way of implementing the generating process would be to draw, at random, pairs of coordinates (i,j) and, again assuming finite memory, to examine the Q points ($0 \leq Q \leq P$) already generated within the neighborhood of point (i,j). Intensity at that point would then be drawn conditionally to the Q already-drawn intensities. But again we would have to "invent" higher-order joint densities. Mathematically the problem can then be stated as follows: given M random variables $X_1,\ldots,X_M$ such that for every pair (X_i,X_j), $1 \leq i,j \leq M$ and $i \neq j$, we know the joint density function $p(x_i,x_j)$, find a function $p(x_1,\ldots,x_M)$ which satisfies

$$p(x_1,\ldots,x_M) \geq 0 \qquad \text{for all } x_1,\ldots,x_M \tag{4}$$

$$\sum_{x_K,\ K\neq i \text{ and } K\neq j} P(x_1,\ldots,x_M) = P(x_i,x_j) \tag{5}$$

The $p(x_i,x_j)$'s are sometimes called the marginals of $p(x_1,\ldots,x_M)$. Assuming quantization with K levels, the K^M unknowns $p(x_1,\ldots,x_M)$ can be stacked up in a vector $\vec{p}$ and conditions (4) and (5) correspond to a linear programming problem:

$$\begin{cases} A\vec{p} = \vec{m} & (6) \\ \vec{p} \geq 0 & (7) \end{cases}$$

where vector $\vec{m}$ is obtained from the functions $p(x_i,x_j)$ and matrix A of size $(\binom{M}{2}K^2)\times K^M$ contains only ones and zeroes. For any reasonable values of M and K, this is a set of linear equations and inequalities of fairly large dimension and usual resolution techniques like the simplex [3] become very quickly limited.

Solution through Algebraic Reconstruction Techniques (ART)

ART was introduced by Gordon, Bender and Herman [11] for solving the problem of three-dimensional reconstruction from projections in electron microscopy and radiology. This is a deconvolution problem where an estimate of a function in a higher-dimensional space is deconvolved from its experimentally measured projections to a lower-dimensional space. For an excellent review of those techniques see Gordon [12].

The problem stated in equations (4) and (5) or (6) and (7) is precisely a problem where the projections are the second-order joint density functions. ART is therefore directly applicable. The simple intuitive basis is that each projected density is thrown back across the higher-dimensional region from whence it came, with repeated corrections to bring each projection of the estimate into agreement with the corresponding measured projection.

Formally, we use an iterative scheme defined by

$$\tilde{p}^{(q+1)}(x_1,\ldots,x_M) = \tilde{p}^{(q)}(x_1,\ldots,x_M)+tc^{(q)}(x_1,\ldots,x_M)$$

$$\text{for all values of } x_1,\ldots,x_M \tag{8}$$

$$\text{for } q = 0,1,\ldots$$

the correction term $c^{(q)}(x_1,\ldots,x_M)$ is given by:

$$c^{(q)}(x_1,\ldots,x_M) = \frac{1}{K^{M-2}\binom{M}{2}} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} (p(x_i,x_j)-\tilde{p}^{(q)}(x_i,x_j)) \tag{9}$$

where $p(x_i,x_j)$ is the actual marginal measured, for example, from an original texture field and $\tilde{p}^{(q)}(x_i,x_j)$ is the marginal corresponding

to the reconstructed density at iteration q.

We may express this in words as follows. The iterative process may be started with all reconstruction elements set to a constant $(p^{(0)}(x_1,\ldots,x_M) = \frac{1}{K^M}$ for all $(x_1,\ldots,x_M)$. In each iteration the sum of the differences between the actual and the reconstructed marginals is computed and evenly divided amongst the K^{M-2} reconstruction elements. If the correction is negative, it may happen that the calculated density becomes negative at some points. This problem can be alleviated by using a modified iteration scheme defined by

$$\tilde{p}^{(q+1)}(x_1,\ldots,x_M) = \text{Max}\{0, p^{(q)}(x_1,\ldots,x_M) + tc^{(q)}(x_1,\ldots,x_M)\} \qquad (10)$$

therefore guaranteeing $\tilde{p}^{q+1} \geq 0$ (constrained ART [12]). It is of course necessary to determine when an iterative algorithm has converged to a solution which is optimal according to some criterion. This is in turn related to the problem of finding the optimal value t_q of t. Various criteria for convergence have been devised [12]. For simplicity, we chose the mean-square error between the measured and calculated marginals:

$$\varepsilon_q = \|\vec{e}^{(q)}\|_2^2 = \|\vec{m} - \vec{\tilde{m}}^{(q)}\|_2^2 \qquad (11)$$

where $\|\cdot\|_2$ is the usual euclidean norm and $\vec{\tilde{m}}$ is defined in equation (6). The derivation is given in the Appendix.

A dual approach that we explored also is based upon the analysis of equation (5) in the Fourier domain. It can easily be shown that the initial problem stated by equations (4) and (5) is equivalent to an interpolation problem in the Fourier domain. The major drawback of this approach is the difficulty to ensure the positivity of the inverse Fourier transform of the interpolated function. Therefore we did not pursue in that direction even though it may be the case that "good" interpolating functions will alleviate that problem.

The basic philosophy of the two approaches we just discussed is that Mth-order joint density functions are "invented" to satisfy exactly the constraints stated in equation (5). Their obvious disadvantage is the high dimensionality (K^M for an Mth-order joint density function) of the data that is to be stored compared with the usually lower dimensionality ($\binom{M}{1}K^2$ for the second-order joint density functions) of the data that is effectively used. An obvious way of minimizing this problem is to relax the constraints of equation (5) and be content with functions $p(x_1,\ldots,x_M)$ which satisfy only a subset of these constraints. Chow's expansion ([13], Chapter 4) paves the way to such an approximation with the advantage, when compared with other expansions such as the Rademacher-Walsh and the Bahadu-Lazarsfeld, that it always provides a full-fledged joint density function, that is positive and summing to 1. Chow's expansion can be written as

$$p(x_1,\ldots,x_M) = p(x_1)p(x_2|x_1)\ldots p(x_M|x_1,x_2,\ldots,x_{M-1}) \tag{12}$$

This is an exact representation and as proposed in [14], product approximations can be found that use only second-order densities. Their general definition is

$$\tilde{p}(x_1,\ldots,x_M) = \prod_{i=1}^{M} p(x_{m_i}|x_{m_{k(i)}}) \qquad 0 \leq j(i) < i \tag{13}$$

where $(m_1,\ldots,m_M)$ is some permutation of $(1,\ldots,M)$, $m = 0$ and $p(x_1|x_0) = p(x_i)$. It can be easily shown that $\tilde{p}$ is a valid density function and that if $k(i)$ equals 0 for only one value of i, equation (5) holds for $M-1$ pairs of indexes i and j. Unfortunately we have not been able to obtain good synthesis using this approach at the present time.

Results and Conclusions

The iterative process defined in (8) may be halted at any number of iterations, q, and a texture may be generated using the value of $\vec{p}$ at that point. However, it should be kept in mind that the success of a texture synthesis depends on making the error $c^{(q)}$ as small as possible and the texture generation process is sensitive to this error. It has also been found by experimentation that the $\vec{p}$ contains many values which are set to zero by implementation of constrained ART (10). This tends to cause the Markov texture generating process to become absorbing, which causes patches of white and black or streaks and lines to be generated. This is eliminated by setting those values which are zero to a small non-zero value, ε, in the generation process.

Using the above concepts, texture simulations of the binary textures water (Figure 1a) and raffia (Figure 1b) were generated (Figures 2a,b). Synthesized textures employing actual kth-order statistics (K=14) were also generated (Figures 3a,b). Simulations of other textures employing kth-order statistics and details of the texture generation method used are contained in [6].

Appendix

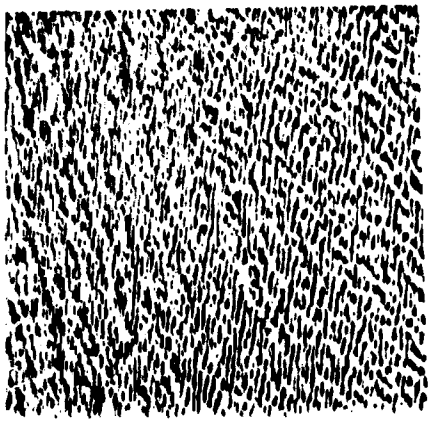
Derivation of t

Rewriting equation (8) in vector form, we get:

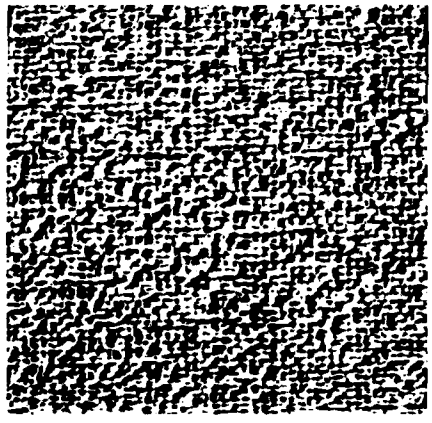
$$\vec{p}^{(q+1)} = \vec{p}^{(q)} + t\vec{c}^{(q)} \qquad (A1)$$

Multiplying both sides of equation (A1) with matrix A (equation (6)) we obtain

$$\vec{m}^{(q+1)} = \vec{m}^{(q)} + t\vec{d}^{(q)} \qquad (A2)$$

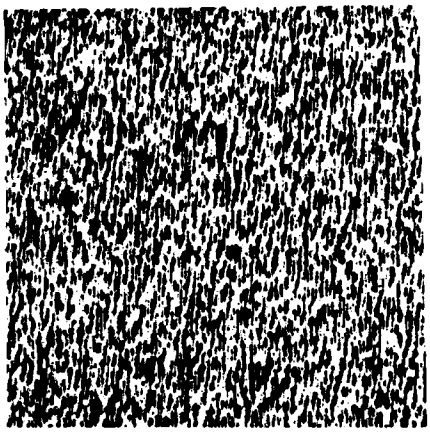


(a)

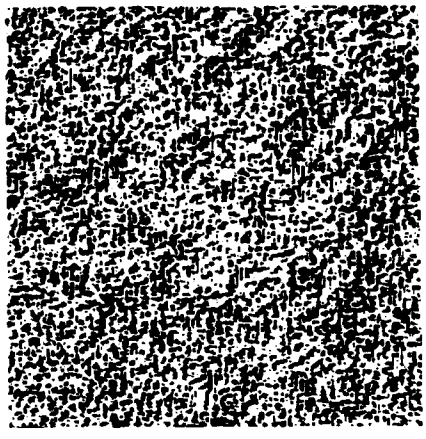


(b)

Figure 1

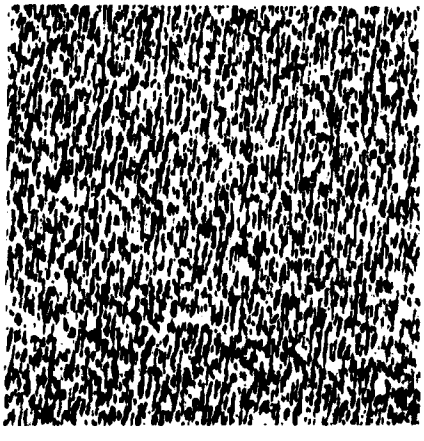


(a)

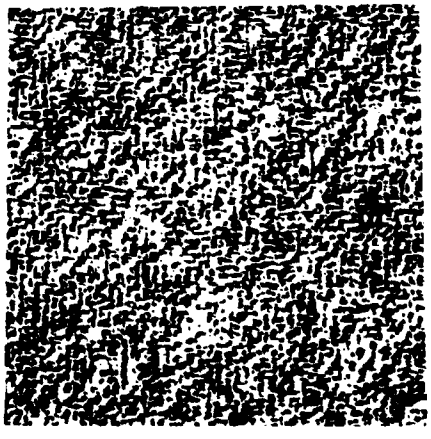


(b)

Figure 2



(a)



(b)

Figure 3

where

$$\vec{d}(q) = A\vec{c}(q) \tag{A3}$$

subtracting the actual marginal vector $\vec{m}$ from both sides of equation (A2) yields

$$\|\vec{e}(q+1)\|_2^2 = \|\vec{e}(q) - t\vec{d}(q)\|_2^2 = t^2\|\vec{d}(q)\|_2^2 - 2t\vec{d}(q)\cdot\vec{e}(q) + \|\vec{e}(q)\|_2^2 \tag{A4}$$

Therefore the error ε_{q+1} at iteration q+1 is minimized for

$$t_q = \frac{\vec{e}(q)\cdot\vec{d}(q)}{\|\vec{d}(q)\|_2^2} \tag{A5}$$

where $\cdot$ denotes the inner product.

References

1. B. Julesz, "Visual Pattern Discrimination," IRE Transaction on Information Theory, Vol. IT-8, No. 1, pp. 84-92, February 1962.

2. B. Julesz, E. Gilbert and J.D. Victor, "Visual Discrimination of Textures with Identical Third Order Statistics," Biological Cybernetics, Vol. 31, pp. 137-140, 1978.

3. A. Gagalowicz, "Stochastic Texture Fields Synthesis from a priori given second order Statistics," Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, pp. 376-381, Chicago, August 6-8 1979.

4. W.K. Pratt, O.D. Faugeras and A. Gagalowicz, "Visual

Discrimination of Stochastic Texture Fields," IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-8, NO. 11, pp. 796-804, November 1978.

5. O.D. Faugeras and W.K. Pratt, "Decorrelation methods of texture feature extraction," to appear in the IEEE Trans. on Pattern Analysis and Machine Intelligence, 1980.

6. D.D. Garber, "Models for Texture Analysis and Synthesis," Technical Report USCIPI 910, 1979.

7. D.D. Garber, "One-Dimensional Texture Pattern Generation and Discrimination," Technical Report USCIPI 840, 1978.

8. B.H. McCormick and S.N. Jayaramamurthy, "Time series model for texture synthesis," Int. J. Comput. Inform. Sci., Vol. 3, No. 4, pp. 329-343, December 1974.

9. J.T. Tou, D.B. Kao and Y.S. Chang, "Pictorial Texture Analysis and Synthesis," presented at Third Int. Joint Conf. on Pattern Recognition (Coronado, Ca.), August 1976.

10. S.Y. Lu and K.S. Fu, "A Syntactic approach to texture analysis," Comput. Graph. Image Processing, Vol. 7, pp. 303-330, 1978.

11. R. Gordon, R. Bender and G.T. Herman, "Algebraic Reconstruction Techniques (ART) for three-dimensional electron microscopy and x-ray photography," J. Theor. Biol., 29, pp. 471-481, 1970.

12. R. Gordon, "A Tutorial on ART," IEEE Trans. on Nuclear Sciences, Vol. NS-21, pp. 78-93, June 1974.

13. R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, Wiley, 1973.

14. C.K. Chow and C.N. Liu, "Approximating Discrete Probability

Distribution with Dependence Trees," IEEE Trans. on Information Theory, Vol. IT-14, No. 3, pp. 462-467, May 1968.

2.7 Autoregresive Modeling with Conditional Expectations for Texture Synthesis

O.D. Faugeras

Introduction

Julesz' conjecture that second order statistics are sufficient in terms of human visual texture discrimination [1] provides a useful bound on the amount of information needed to reconstruct a texture viewed as a realization of a random process. Even though this conjecture has recently been disproved [2,3], it nonetheless provides an excellent first order description of human texture perception. For an investigation of the tightness of that bound see [4] and for an application to texture analysis see [5]. We show in a companion paper [6] that even with this upper bound an approach to synthesis based upon the use of high order density functions is rapidly limited by the size of the data to be handled. On the other hand, an approach based only upon the linear prediction of the intensity value at one pixel from neighboring pixels intensities, although quite simple, takes only into account the autocorrelation function of the texture. Early work on the subject by McCormick and Jayaramurthy [7] made use of a one-dimensional autoregressive model. Deguchi and Morishita [8], Tou et al. [9] and Tau and Chang [10] also used a similar technique. Garber [11] extended these ideas to a two-dimensional model. Obviously, except for binary fields or gaussian fields this is a lot less information than what is contained in second order statistics. These two reasons prompted us to explore an intermediate approach in

which we linearly predict the intensity value of one pixel from the expected values of that intensity given the intensity values of sets of neighboring pixels. The advantages of this approach are that high order statistics can be taken into account in the synthesis process in a controlled manner while keeping the simplicity of the linear prediction techniques.

We will describe in this paper a method suitable for analyzing and synthesizing so called "natural" texture fields where the word natural means that there is a sufficient amount of randomness so that a modelization based upon stochastic fields theory is not completely preposterous. We therefore eliminate from our scope purely repetitive patterns probably best described by a structural approach.

The Model

Let $x_{m,n}$ be a discrete stochastic field where integers m and n are the space coordinates and $x_{m,n}$ is the intensity at point (m,n). For convenience in the notations we will use only one index k to denote the pair (m,n), k may therefore be thought of as a vector of coordinates (m,n).

Linear prediction techniques attempt to approximate the random variable x_k by a weighted sum of the random variables at neighboring points.

$$\tilde{x}_k = \sum_{i=1}^{N} \alpha_i x_i \qquad (1)$$

The α_i's are then adjusted to minimize the variance of the error

$$E\{(x_k - \tilde{x}_k)^2\}$$

where $E\{\cdot\}$ denotes the expected value. This is achieved simply by

solving a set of linear equations

$$A\vec{\alpha} = \vec{b} \tag{2}$$

where matrix A is NxN symmetric with $a_{ij} = E\{x_i x_j\}$, $\vec{\alpha} = [\alpha_1,\ldots,\alpha_N]^T$ and $\vec{b} = [E\{x_k,x_1\},\ldots,E\{x_k,x_N\}]^T$. Therefore, the α_i's contain only information about the autocorrelation function of the stochastic field.

Our attempt to use more than second order moments while keeping the simplicity of the linear prediction approach is based on using the expectations of the value at point k conditioned on the values at neighboring points. Very simply, if we know the random variables $x_{i_1},\ldots,x_{i_q}$, the best prediction of x_k is

$$E\{x_k | x_{i_1},\ldots,x_{i_q}\}$$

In other words $E\{x_k | x_{i_1},\ldots, x_{i_p}\}$ is the function of $x_{i_1},\ldots,x_{i_p}$ which provides the best approximation of x_k in the mean-square sense [12]. Thus, we define

$$\tilde{x}_k = \sum_{i=1}^{K} \alpha_i E\{x_k | y_i\} \tag{3}$$

as as estimate of x_k, where y_i denotes the q-tuple $(x_{i_1},\ldots x_{iq})$ of random variables at nearby points. Notice that computing $E\{x_k | y_i\}$ implies that we know the q+1st order joint density function $p(x_k,y_i)$. The choice of q and K is governed by our willingness to achieve a more accurate prediction. Once this choice has been made, the α_i's are computed so that they minimize the error variance

$$E\{(x_k-\tilde{x}_k)^2\}$$

The mathematics of the derivation are essentially the same as in the

simple case of the linear prediction technique. The details are given in Appendix A. We again have to solve a set of linear equations.

$$A\vec{\alpha} = \vec{b} \tag{4}$$

A is a KxK symmetric matrix whose current element a_{ij} is given by

$$\begin{aligned} a_{ij} &= E\{E\{x_k|y_i\}E\{x_k|y_j\}\} \\ &= \sum_{y_i,y_j} E\{x_k|y_i\}E(x_k|y_j)p(y_i,y_j) \end{aligned} \tag{5}$$

vector $\vec{b}$'s ith coordinate is

$$b_i = E\{x_k E\{x_k|y_i\}\} = \sum_{x_k,y_i} x_k E\{x_k|y_i\}p(x_k,y_i) \tag{6}$$

$p(y_i,y_j)$ is the (2q)th order joint density function of the random variables $(x_{i_1},\ldots,x_{i_q})$ and $(x_{j_1},\ldots,x_{j_q})$ while $p(x_k,y_i)$ is the q+1st order joint density function of the random variables x_k and $(x_{i_1},\ldots x_{i_q})$.

Although everything we said is valid in the continuous case where the random variables x_k can take any values we deal in practice with quantized values. We will therefore assume in what follows that the x_k's take only G integer values (grey levels) which we can assume without restriction of generality to be 0,1,...,G-1.

Complexity, Synthesis Method

The model contained in Eq. (3) is made of two different parts. First the conditional expectations $E\{x_k|y_i\}$ have to be computed from some real data by estimating the q+1st order joint density function

$p(x_k, y_i)$

$$E\{x_k|y_i\} = \sum_{x_k=0}^{G-1} x_k p(x_k|y_i) \tag{7}$$

There are G^q values to compute and store. Second, numbers α_i are computed by solving Eq. (4). This implies estimating the (2q)th order joint density functions. $p(y_i, y_j)$ for $i = 1,\ldots,K$; $j = i+1,\ldots,K$ each one being represented by G^{2q} values. The texture is then represented by the KxG^q numbers $E\{x_k|y_i\}$ and the K numbers α_i.

We adopted for simplicity a synthesis method based on a top to bottom, left to right scanning of the image. Therefore, the actual neighborhood used to predict the value x_k at point k is as indicated in Fig. 1. At this point it may be worthwhile to discuss problems which we encountered when we tried to implement these ideas. The first one is related to the model that we used. In order to generate an actual texture field (for example a 512x512 image) the process described by Eq. (3) has to be seeded. For points in the image located close to the first row or the first and last columns intensity values have to be set at neighboring pixels lying outside the image. These intensities are usually taken as values obtained from a pseudo-random number generator. But even with such random initial conditions, the process described by Eq. (3) converges quickly toward a deterministic steady state solution. To avoid that problem (also encountered in [5]) we can rewrite Eq. (3) as

$$\tilde{x}_k = \sum_{i=1}^{K} \alpha_i E\{x_k|y_i\} + \beta u_k \tag{8}$$

where the process u_k is a white process composed of zero mean, unit variance equidistributed random variables independent of the random field x_k. Because of these hypothesis nothing changes in the

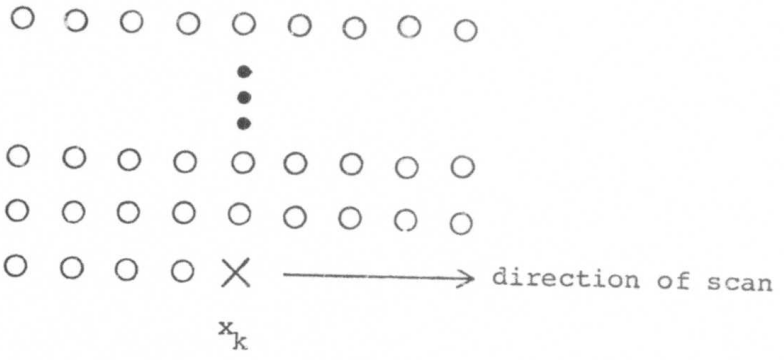
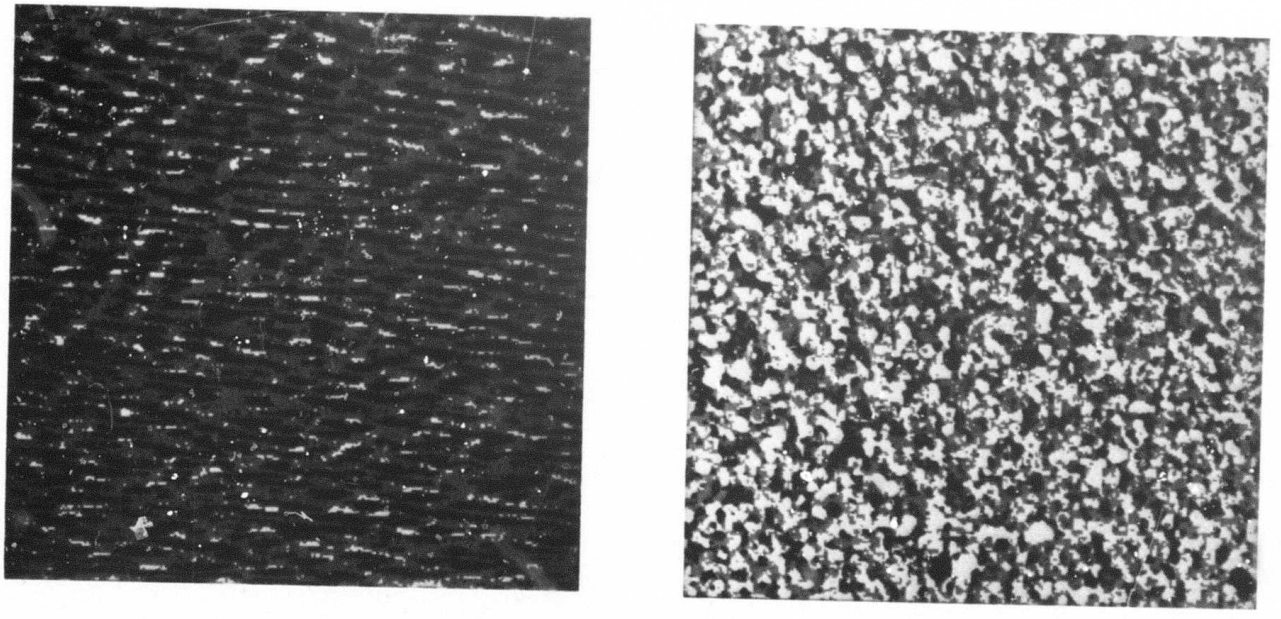


Figure 1. Neighborhood used to predict the value x_k



a).

b).

Figure 2. Original Textures: a) Water and b) Sand.

determination of the numbers α_i but the parameter can be adjusted so that the variance of $\tilde{x}_k$ is the same as the variance of x_k (see Appendix B).

The second problem that we encountered is related to an inherent hypothesis that is always made when dealing with images modeled as random fields, the ergodicity assumption. This in turn implies that the process is stationary and therefore that its statistics are invariant with translations in the plane. This forces some non trivial constraints upon the different nth order joint density functions. Without going into two much details (for a somewhat more precise description see [3]), let us look at the simple case of the second-order joint density functions $p_{ij}(x_i,x_j)$. They must satisfy

$$\sum_{x_i=0}^{G-1} p_{ij}(x_i,x_j) = p(x_j) \tag{9}$$

and

$$\sum_{x_j=0}^{G-1} p_{ij}(x_i,x_j) = p(x_i) \tag{10}$$

for all (i,j) of pixels ($i \neq j$). p(x) is the first order density of the process (the same at every point because of the stationarity). For higher order statistics, more complicated relations must hold. Since these statistics have to be estimated from one texture sample of finite size (using the ergodicity assumption) special care has to be taken in order to make sure that constraints such as Eqs. (9) and (10) are actually verified.

Experimental Results

We have applied our algorithm to the synthesis of two natural texture fields "Water" and "Sand" shown in Figure 2. These textures are quantized with four grey levels (G=4) and second order joint density functions (q=2) are estimated in a 15x15 neighborhood (K=217). Results of the synthesis are shown in Figure 3. We plan to experiment with more textures and different values of the parameter G, q and K in the very near future.

Appendix A

We want to minimize:

$$\mathcal{E}_2 = E\{(x_k - \tilde{x}_k)^2\} \tag{A-1}$$

with respect to the variables α_i, where $\tilde{x}_k$ is given by Eq. (3). Setting $\frac{\partial \mathcal{E}_2}{\partial \alpha_i}$ equal to 0 for i=1,...,K yields

$$E\{E\{x_k|y_i\}(x_k - \sum_{j=1}^{K} \alpha_j E\{x_k|y_j\})\} = \emptyset \tag{A-2}$$

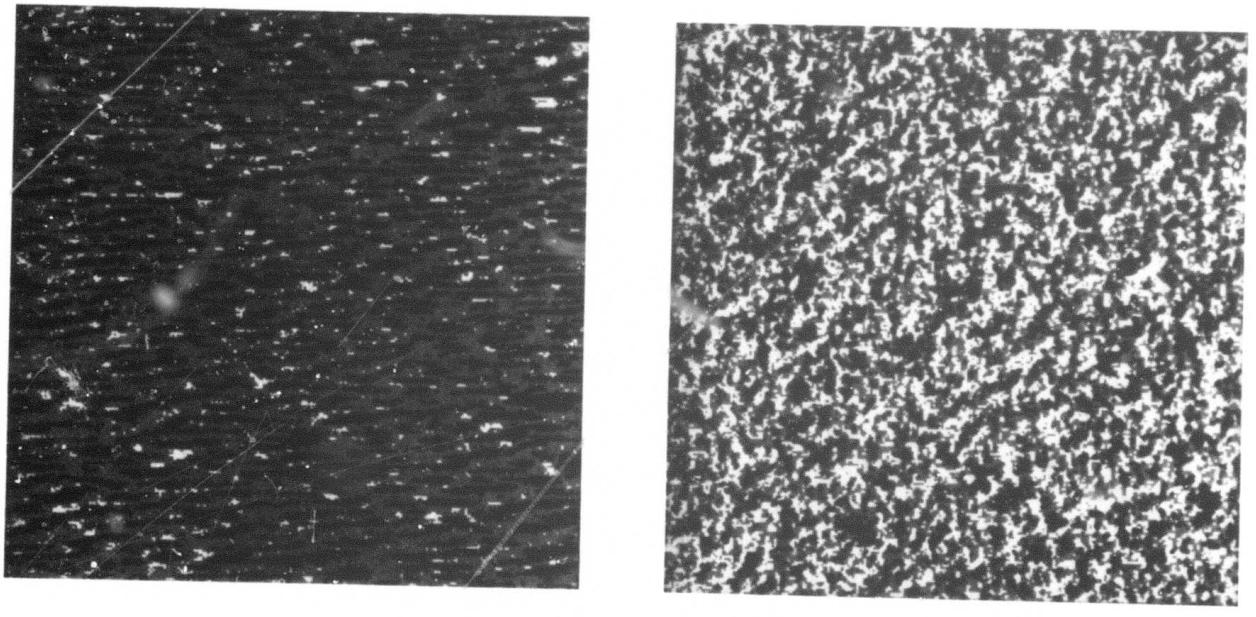
and therefore

$$\sum_{j=1}^{K} \alpha_j E\{x_k|y_j\} = E\{x_k E\{x_k|y_i\}\} \tag{A-3}$$

for i=1,...,K.

Appendix B

Let μ and σ^2 be the mean and variance of x_k. If $\tilde{x}_k$ is given by Eq. (8), β is adjusted to minimize



a).

b).

Figure 3. Synthetic Textures: a) Water and b) Sand
G = 4, q = 1, K = 217.

$$\delta_2' = (E\{(\tilde{x}_k - \bar{\tilde{x}}_k)^2\} - \sigma^2)^2 \tag{B-1}$$

where $\bar{\tilde{x}}$ denotes the mean of $\tilde{x}_k$. It is readily shown that

$$E\{\tilde{x}_k^2\} = \beta^2 + \vec{\alpha}^T A \vec{\alpha} = \beta^2 + \vec{\alpha}^T \cdot \vec{b} \tag{B-2}$$

$$\bar{\tilde{x}}_k = E\{\tilde{x}_k\} = \mu \sum_{i=1}^{K} \alpha_i \tag{B-3}$$

and therefore

$$\delta_2' = (\beta^2 + \vec{\alpha}^T \cdot b - \mu^2 (\sum_{i=1}^{K} \alpha_i)^2 - \sigma^2)^2 \tag{B-4}$$

if the quantity $\vec{\alpha}^T \cdot \vec{b} - \mu^2 (\sum_{i=1}^{K} \alpha_i)^2 - \sigma^2$ is positive, δ_2' is minimized for $\beta = 0$ otherwise for

$$\beta^2 = \alpha^2 + \mu^2 (\sum_{i=1}^{K} \alpha_i)^2 - \vec{\alpha}^T \cdot \vec{b} \tag{B-5}$$

References

1. B. Julesz, "Visual Pattern Discrimination," IRE Transactions on Information Theory, Vol. IT-8, No. 1, pp. 84-92, February 1962.

2. B. Julesz, E. Gilbert, and J.D. Victor, "Visual Discrimination of Textures with Identical Third Order Statistics," Biological Cybernetics, Vol. 31, pp. 137-140, 1978.

3. A. Gagalowicz, "Stochastic Texture Fields Synthesis from A Priori Given Second Order Statistics," Proc. of the IEEE Computer Society

Conference on Pattern Recognition and Image Processing, pp. 796-804, Chicago, August 6-8, 1979.

4. W.K. Pratt, O.D. Faugeras, and A. Gagalowicz, "Visual Discrimination of Stochastic Texture Fields," IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-8, No. 11, pp. 796-804, November 1978.

5. O.D. Faugeras and W.K. Pratt, "Decorrelation Methods of Texture Feature Extraction," IEEE Trans. on Pattern Analysis and Machine Intelligence, In press, 1980.

6. O.D. Faugeras and D.D. Garber, "Algebraic Reconstruction Techniques for Texture Synthesis," submitted to the 5th International Conference on Pattern Recognition.

7. B.H. McCormick and S.N. Jayaramamurthy, "Time Series Model for Texture Synthesis," Int. J. Comput. Inform. Sci., Vol. 3, pp. 329-343, December 1974.

8. K. Deguchi and I. Morishita, "Texture Characterization and Texture-Based Image Partitioning Using Two-Dimensional Linear Estimation Techniques," presented at U.S.-Japan Cooperative Science Program Seminar on Image Processing in Remote Sensing (Washington, D.C.), Nov. 1-5, 1976.

9. J.T. Tou, D.B. Kao, and Y.S. Chang, "Pictorial Texture Analysis and Synthesis," Third Int. Joint Conf. on Pattern Recognition (Coronado, CA.), Aug. 1976.

10. J.T. Tou and Y.S. Chang, "An Approach to Texture Pattern Analysis and Recognition," in Proc. 1976 IEEE Conf. on Decision and Control, 1976.

11. D.D. Garber, "Models for Texture Analysis and Synthesis," Technical Report USCIPI 910, 1979.

12. A. Papoulis, Probability, Random Variables and Stochastic Processes, McGraw-Hill, 1965.

2.8 Higher-Order Texture Synthesis Models and Residual Examination

D.D. Garber and A.A. Sawchuk

Introduction

Earlier work [1] has shown the usefulness of the Markov and linear autoregressive model in the analysis and synthesis of natural textures. Each model was assumed to be the random process used to generate a texture. Then, based on this assumption, model parameter estimates were made using data from a parent texture and these parameter estimates were then used to generate simulations of that texture using the assumed model. The linear model was developed to add simplicity to the earlier Markov model which requires a large number of storage locations for generation parameters. The simplification arising from such a model reduction naturally causes information to be lost, resulting in the generation of less-appealing textures. It may be possible to regain some lost information by adding important parameters to the reduced model, thus increasing its complexity.

Markov and First-Order Models

Texture generation work up to the present [1] has been based on the concept of generating a pixel, V_{N+1}, on a line of an image based on the values of previous pixels V_i which are located above or to the left of that pixel (Figure 1). The general Markov-N model does this

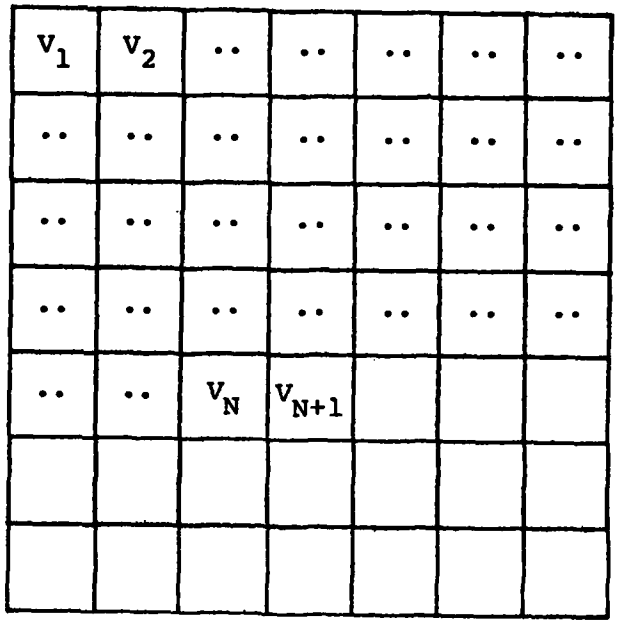


Figure 1. Two-Dimensional Synthesis.

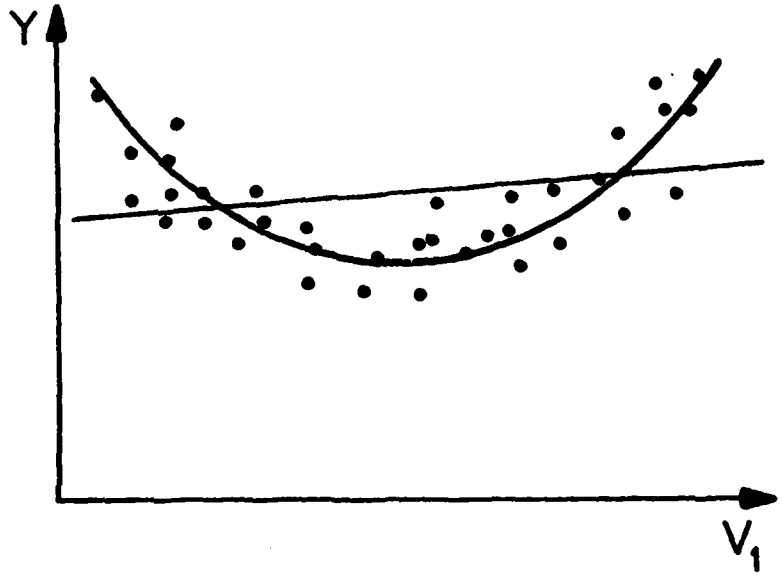


Figure 2. Quadratic versus linear fit.

by using a set of generation parameters

$$G_{V_{N+1}}(V_1,V_2,\ldots,V_N) \tag{1}$$

where

$$G_{V_{N+1}}(V_1,V_2,\ldots,V_N) = P(V_{N+1}|V_1,V_2,\ldots,V_N)$$

and P(A|B) represents the conditional probability of A given B. Using this model, for each set of V_i, i=1,...,N we have a unique distribution of V_{N+1} which can assume any form. Note that the set of parameters (1) not only determines the mean and standard deviation of the next pixel V_{N+1} but also all higher moments. The limitation of this model is realized rapidly when it is understood that g^{N+1} of these parameters must be used to generate a texture as each unique set of V_i, i=1,..,N+1, requires the storage of a generation parameter where g is the number of distinct grey levels a pixel can assume and N is the number of pixels in the generation kernel to be used in the generation process.

The less-complex linear model [1], also known as an autoregressive model,

$$Y = X\beta+\varepsilon \tag{2}$$

where $Y = V_{N+1}$ $\quad X^T = \begin{bmatrix} V_1 \\ V_2 \\ \cdot \\ \cdot \\ \cdot \\ V_N \\ 1 \end{bmatrix}$

was then proposed to generate textures. In this case each V_{N+1} depends on a linear combination of the previous V_i and some unexplained variance ε. In this case, only N parameter estimates are

required. (N coordinate points are also required if the kernel is non-contiguous [2]). The reduction of a model containing g estimates to one containing only N will naturally cause a loss of information to occur. Geometrically speaking, we are approximating an (N+1)-dimensional surface with an N-dimensional hyperplane. In the linear model case, we obtain an estimate of the mean of V_{N+1} rather than its complete distribution for a given set of V_i, i=1,...,N. Randomness used in the generating process involving this model was obtained by assuming that ε has some fixed distribution with zero mean. The extent to which the hyperplane approximates the (N+1)-dimensional surface and the degree to which the eye can detect this difference will determine an observer's ability to discriminate between a texture generated by the Markov model versus the linear one.

The degree to which the Markov model may be used to explain natural texture has been demonstrated only in the case of binary textures. In that case, the model worked well, exhibiting short-comings mainly in cases where irregular macro-structure, exceeding the size of the neighborhood of V_i's used, was present. Application of this model to continuous-tone textures remains to be undertaken and is a rather large task in itself due to the size and amount of data to be stored and utilized in the form of generation parameters.

Second-Order Model

When we say that a model is linear or nonlinear, we are referring to linearity or nonlinearity in the parameters. The value of the highest power of an independent variable in the model is called the order of the model. For example,

$$Y = \beta_1 V_1 + \beta_{11} V_1^2 + \beta_0 + \varepsilon \qquad (3)$$

is a second-order linear model. A general second-order linear model with two independent variables may be written as

$$Y = \beta_1 V_1 + \beta_2 V_2 + \beta_{11} V_1^2 + \beta_{22} V_2^2 + \beta_{12} V_1 V_2 + \beta_0 + \varepsilon \quad (4)$$

A full second-order model with K independent variables will employ $(K^2+3K)/2$ terms in addition to the β_0(constant) and ε (error) terms. Second-order models have been particularly useful in studies where surfaces must be approximated by polynomials of low order. In all cases, a second-order model will "fit" given data as well as or better than a first-order model containing the same independent variables as the first-order model is a subset of second-order models. This does not imply that the second-order model will be more correct however as the process which we are attempting to explain may be in fact a linear first-order process or some other type.

It is also important to note that the covariance of the V_i are required in order to obtain least-square estimates of the parameters β_i in the first-order model [2]. Covariance is essentially a second-order statistic on a set of data. Therefore, estimating the parameters of a second-order model will require the use of fourth-order statistics. Specifically the correlation of terms $V_{i_1} V_{i_2}$ and $V_{i_3} V_{i_4}$ will be utilized. This may cause serious problems as in may cases the variables in a second-order model will be highly intercorrelated. For example, the terms V_1, V_1^2 and $V_1 V_i$ (if V_i is highly related to V_1) may be strongly correlated. This situation, often referred to as multicollinearity, may cause problems during the inversion of the estimated correlation matrix, a necessary step in model parameter estimation. For this reason, care should be exercised during the analysis of second-order models.

The use of a second-order model to approximate the surface of the general Markov model could have many advantages over a first-order model. An example of fitting such a model in one dimension to a given set of data is shown in Figure 2.

Still the linear first-order model may provide a good fit to the data and the magnitude of the unexplained variance in the data may be large enough that the improvement due to the addition of second-order terms to the model may be barely noticeable. In two dimensions, the fitting problem is one utilizing a quadric surface such as an elliptic paraboloid or hyperbolic paraboloid versus a plane to fit a given set of data. Again, the fit may or may not be markedly better.

An Example

In order to test whether the addition of second-order terms to the linear model would improve the quality of generated textures, the texture sand (Figure 3) was chosen. It was felt that the linear model containing only first-order terms failed to accurately simulate this texture (Figure 4) and that improvement could be made using a second-order model.

In this case a kernal size of N=70 neighbors was chosen based on ideas discussed in [2] for the first-order model case. That is, each pixel generated depended on the linear combination of 70 pixels above or to the left of it. When adding crossterms to the model it is most important keep in mind that for a neighborhood of N points, $N^2/2$ second-order terms must be considered for possible entry into the second-order model. For N=70 this is 2450 crossterms. As it is numerically infeasible to invert a (2450+70)x(2450+70) matrix a search procedure known as stagewise regression was used. Basically, one uses the best first-order linear model which is a linear function of N pixels in a neighborhood and calls that the new variable z_1. Then as many crossterms as possible are entered into the model as variables z_i, $i=2,\ldots,N_s$, where N_s is a more manageable figure (less than 100). Model parameter estimates are obtained at this step and the best $N_t<N_s$, (those chosen for entry into the model according to some criterion [2]) are kept in the model while the remaining N_s-N_t are discarded. At the next step, N_s-N_t crossterms which have not been previously tested are examined for entry into the model. Again, the

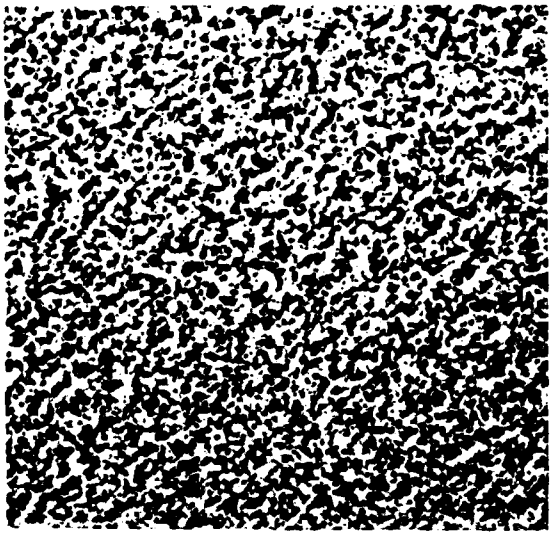


Figure 3. Original SAND.

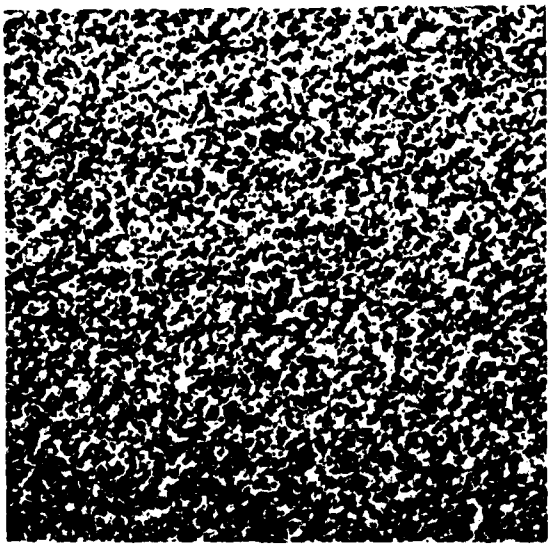


Figure 4. Simulated SAND without cross products.

best N are chosen and the rest are discarded. The process continues until all possible crossterms of interest have been tested for entry into the model. Keeping N_t large helps to insure that valuable model terms are not discarded during the process. Keeping N small, however, reduces the number of steps to be performed, as it allows the examination of more new terms at each step. Each step involves the computation and inversion of a large covariance or correlation matrix. Thus there is a trade-off to be made. This procedure does not provide a true least-squares solution for the variables and cross-terms entered into the final model. Still the solution is a very good one in most cases and is probably the best available given the dimensionality constraints of our problem.

An analysis of this type was performed on the sand texture for N=70. All cross-products containing these points were investigated. The results of the synthesis using this second-order model incorporating white gaussian noise with mean zero and fixed variance to simulate the error term of the model, ε, are shown in Figure 5. The results show only a slight, almost undetectable improvement-much less than desired. This would indicate that the improvement of texture synthesis due to the addition of second-order terms into the linear model is minimal in this case.

Examination of the Residual Image

A residual image is that image which is formed by the differences $\hat{V}_{N+1}-V_{N+1}$ where V_{N+1} is an observed pixel value and $\hat{V}_{N+1}$ is the corresponding fitted value obtained by use of the linear model (2). We can see that these residuals are the differences between what is actually observed and what is predicted by the model, that is, the amount of variation which the linear model has not been able to explain. The usual assumption about the errors, ε, in our model (2) is that they are independent with zero mean and constant variance. An additional assumption of normality allows us to make F-tests when checking variables and crossterms for significance [2]. If our fitted

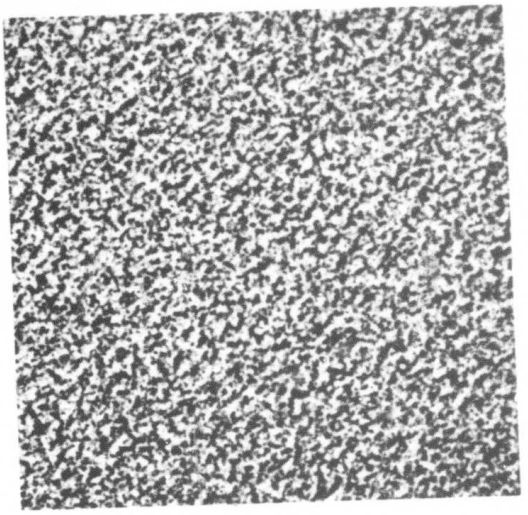


Figure 5. Simulated SAND with cross products.

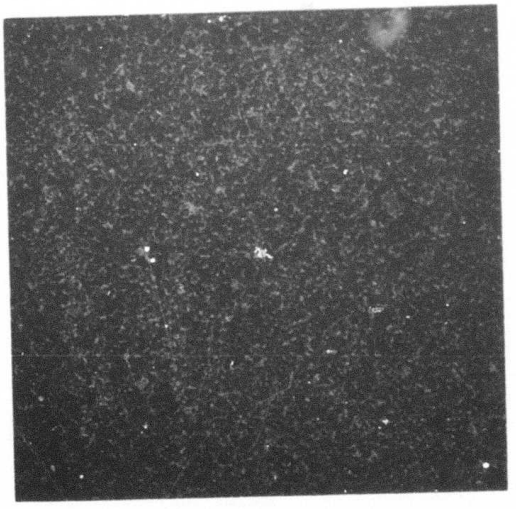


Figure 6. Residual image.

model is correct, the residuals should exhibit tendencies that confirm these assumptions or at least should not exhibit a denial of the assumptions.

A time or space sequence plot of residuals is most fitting in the case of two-dimensional texture synthesis. This is essentially an image containing the pixel differences or residuals $\hat{V}_{N+1}-V_{N+1}$. Naturally we would expect these errors to be small as merely subtracting one pixel from its nearest neighbor would yield a small value in most natural, low-noise images. Such an image of residuals was generated for the sand and linearly rescaled to show the detail present in the image (Figure 6). Definite patterns are seen to exist in this image and thus a violation of the independence assumption is indicated. Ideally, this residual image would be uncorrelated noise.

A histogram showing the number of $\hat{V}_{N+1}$ occurring at each pixel value is shown in figure 7. A plot indicating the mean of the residuals $\hat{V}_{N+1}-V_{N+1}$ versus V_{N+1} is shown in figure 8. As would be expected, residuals where the $\hat{V}_{N+1}$ is less than 0 will have a mean less than zero and those residuals where the V_{N+1} is greater than 255 will be likewise positive. Figure 9 shows a similar plot of the standard deviation of the residuals versus $\hat{V}_{N+1}$. These three figures seem to indicate that the distribution of the error in the model is related to the value $\hat{V}_{N+1}$. Therefore the assumption of constant error variance is questionable. It may be reasonable to drive the generation process with noise which does not have stationary mean or variance. The effect of such a change in the generation process remains to be studied. It is also not yet clear what effect such a change will have on a texture synthesis. Figure 10 shows the distribution of error $\hat{V}_{N+1}-V_{N+1}$ (a histogram of the residual image).

Given the apparent violation of assumptions concerning the model we must re-examine whether these violations may be explained in some specific way and whether they can be adjusted for in a proper manner. Such adjustments may involve transformations of existing variables,

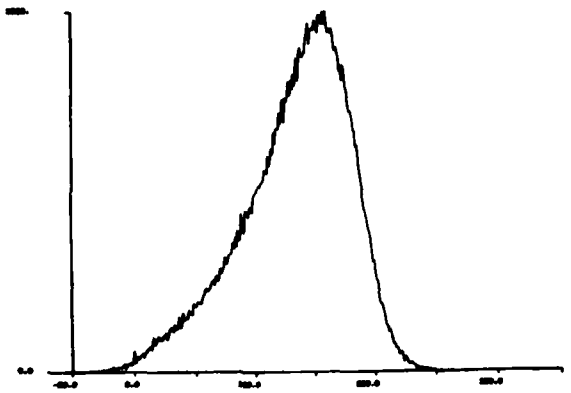


Figure 7. Histogram of $\hat{V}_{N+1}$.

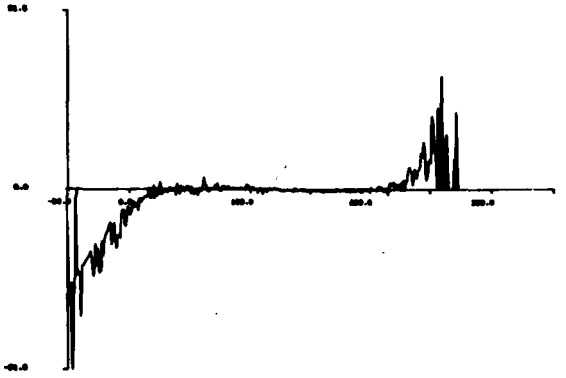


Figure 8. Mean of $\hat{V}_{N+1}-V_{N+1}$ vs $\hat{V}_{N+1}$.

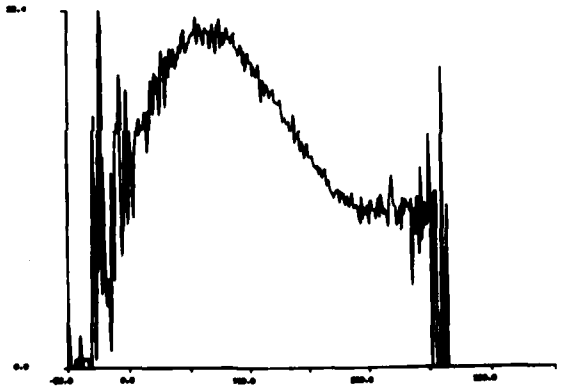


Figure 9. Standard deviation of $\hat{V}_{N+1}-V_{N+1}$ vs $\hat{V}_{N+1}$.

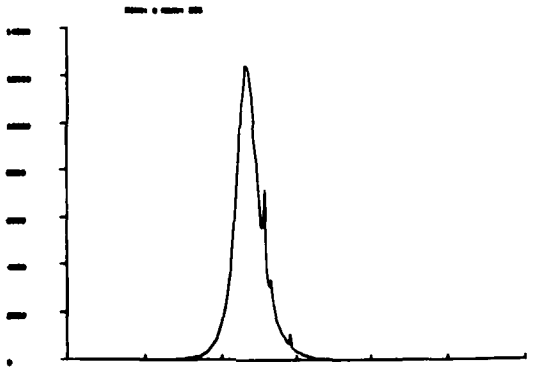


Figure 10. Histogram of residual image.

the generation of new theory concerning texture synthesis or the use of non-stationary non-random noise in the generation process. These ideas and others remain to be investigated.

Conclusions

More investigation into the effect of the addition of second-order terms into the texture model must be done. More conclusive results will only be developed after this study has been performed on a variety of textures. The study of the residual image indicates that many important model assumptions are being violated and thus, at least in some ways, our model may be inadequate. Particularly, in the case of sand, the texture generation method fails to generate irregularly shaped sharp edges, therefore it may be beneficial to propose changes into our model which will induce these edges.

References

1. D.D. Garber, "Models for Texture Analysis and Synthesis," Technical Report USCIPI 910, 1979.

2. D.D. Garber, "Application of the General Linear Model to Binary Texture Synthesis," USCIPI 960, 1980.

2.9 Computer Analysis of Moving Images

B. Bhanu and O.D. Faugeras

Introduction

In the past image understanding systems capable of doing simple image description tasks for a narrow class of images have been constructed. The development of systems for the purposes of detections, identification, localization and tracking the objects through a sequence of images will be a very desirable and important features of such machine vision systems. For example, in the sequence the objects of interest may be the enemy targets tracked by the imaging missile seekers which incorporate sophisticated signal processing to track the targets, aircrafts or blood cells etc. The constituent of the scene may be moving in two or three dimensional space against a stationary foreground/background and sequence of images may be in the optical or some other band of the spectrum. The task at hand is to consider the incorporation of various diverse sources of knowledge that lead to the science of image understanding. Recently both temporal and spatial sequences of images have been used in understanding and analyzing a wide variety of processes in physics, biology, medicine, meterology, navigation etc. [1-5]. Special purpose systems have been built to record and store the sequences of television frames and to correlate information from two cameras as well as from a sequence of images [5,6].

Most of the work in the field of image processing has been concerned with a single frame, where the primary objective is to develop methods to detect, represent, store and manipulate the spatial features of a scene. Although a sequence of images is basically a sequence of static images with a given time function relating the order and time interval between the images of the sequence, yet in analyzing the sequence of images, not only information must be

extracted from each frame, but also from the sequence as such so that a description of the sequence can be obtained. Noise, occlusion, perspective changes of objects, appearance of new objects, disappearance of old objects, shape changes further complicate the tracking and matching problem.

In this paper we consider a new dynamic image understanding system for analyzing the behavior of 2-D curvilinear objects through a temporal sequence of images. Earlier model based systems are principally hierarchical image understanding systems dealing with polygonal shaped objects [8] or curvilinear objects [2,9] or the objects from blocks world. They suffer a lack of adaptability to the task at hand, require more processing and consider only few objects whose number remain fixed in the sequence. The larger number of objects exhibit more complex interactions among the objects and the disappearance and appearance of objects through the boundaries of the images enhances the difficulty of the problem and require the control on symbols and features of the objects in a feedback loop.

The next section describes in brief the previous work, followed by a description of the system for recording the sequences and types of scenes. Finally, we consider the dynamic image understanding model for analyzing the motion and shape of the constituent of the images.

Previous Work

At present digital as well as optical image processing methods are being used for the motion analysis purposes. Both signal and symbolic approaches are under investigation. Originally, the work was stimulated by the interest of analyzing the movement of clouds from a sequence of satellite images. Work has proceeded from there to the analysis of abstract patterns in two dimensions and the analysis of scenes from the blocks world [2,3,5]. Variety of techniques have been used by the researchers working on the interframe coding for comparing image frames, segmenting them into stationary and non-stationary areas

and estimating the translational velocity of moving object. A number of similar techniques have also been used by the researchers in the medicine, biology, industrial automation, robotics, and other related areas. Research efforts involved in the analysis techniques for image sequences can be classified into the following categories.

A. Techniques based on cross-correlation
B. Centroid matching
C. Change detection and Image differencing
D. Dynamics scene modeling
E. Velocity Estimation techniques
F. Hough Transform techniques
G. Optical methods and tracking
H. Moving target indication filter
I. Kalman filtering techniques

These techniques have been discussed in detail in [2,3,7].

System Configuration and Input Scenes

1. System Configuration:

For acquiring both real and synthetic sequences of images, a real time video acquisition and digital display system shown in Fig. 1 has been used. The system was built at the USC Image Processing Institute recently [6]. The system under the control of the PDP 11/34 (RT-11 Operating system) processor will acquire, digitize and store a number of consecutive subframes of video images from the monochrome TV camera. During and after the acquisition phase, the acquired image can be displayed on the monochrome TV monitor. After the acquisition phase, the acquired image subframes can be transferred from the image memory (512x512x8 bits) to the PDP 11/34. An image or image subframes can also be transferred from the PDP 11/34 to the image memory. The maximum size of the image subframe (window area) is 140x400 pixels. The minimum sampling interval between two subframes is of the order of

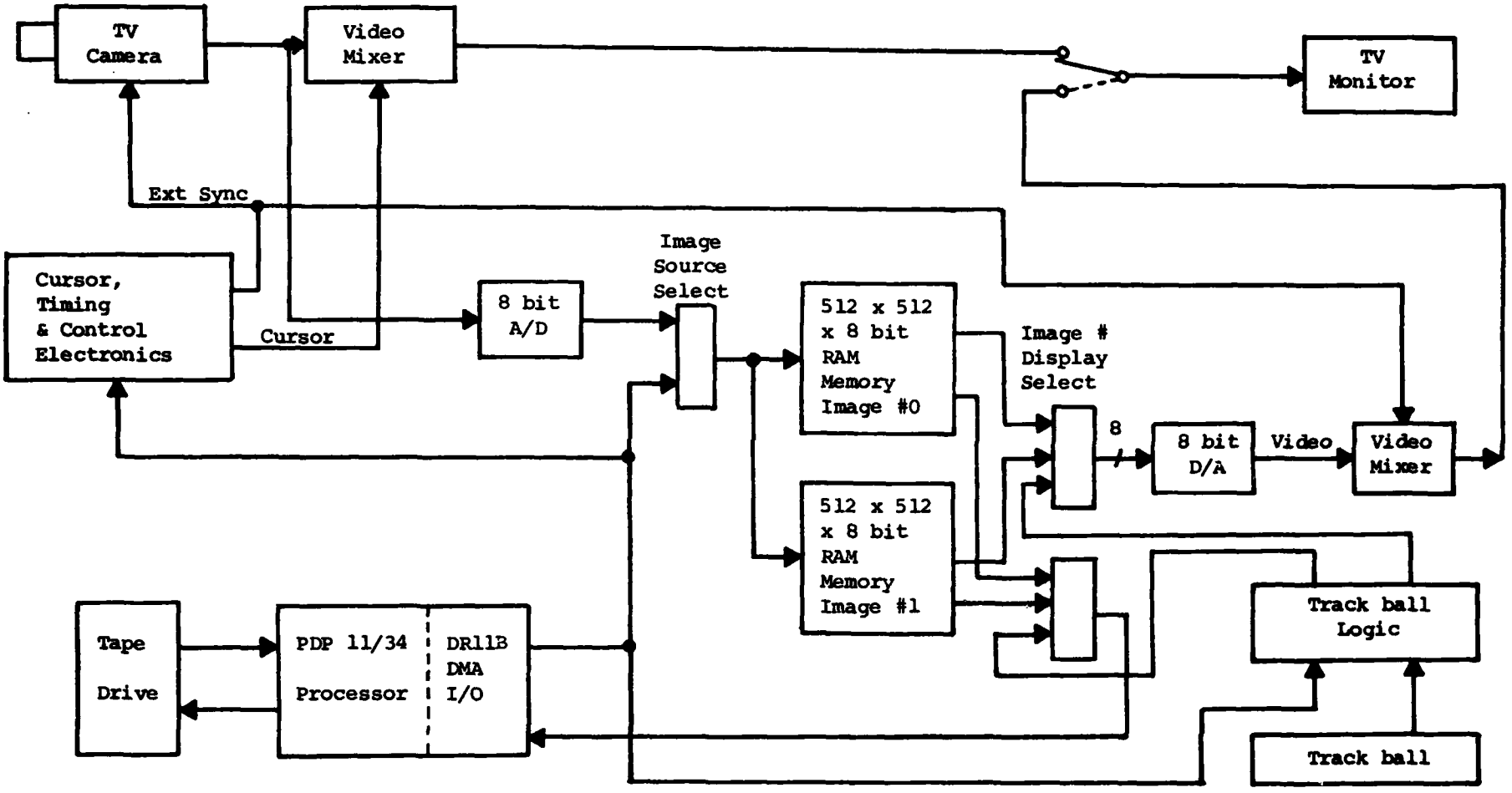


Figure 1. Real time video acquisition and digital display system

0.75 seconds. It is limited by the size of the subframe and the amount of time required to transfer and store the image into a file on the disk or tape.

Programs have been written by authors on PDP 11/34 for the system shown in Fig. 1, which allow the user to store the subframes of specific sizes at a specified rate. Odd and even lines in the subframe can be stored under separate files on the disk for the interlacing to be carried out later on or they can be interlaced before storing it on the disk. Images so stored can be put back on the display under the control of PDP 11/34 for observing a part of the sequence. Sequences of the images obtained are transferred on PDP-10 (Tenex operating system) via a format conversion from RT-11 to RSX-11 and further processing is done on Tenex.

2. Input Scenes

Using the system described above we have generated a number of synthetic and real sequences.

Synthetic sequences exhibit the motion of planar curvilinear two-dimensional objects against the different types of background. The complexity of the scene in these sequences vary from 8 to 15 objects. For generating these sequences the following considerations have been taken into account.

1) Linear and rotational motion of the objects is allowed. Some of the objects may not move at all, while some others may show random behavior.
2) No more than two to four objects may occlude each other at anytime and no more than two to three objects may appear or disappear through the boundaries. At the most an object may divide itself into two objects. Thus the number of objects may not remain fixed throughout the sequence.
3) Objects don't have holes and slow changes in the shape of objects and background are allowed.

4) Changes are slow from frame to frame. They are not drastic or discontinuous. Of course, significant changes may or may not be present between the two frames in the sequence.

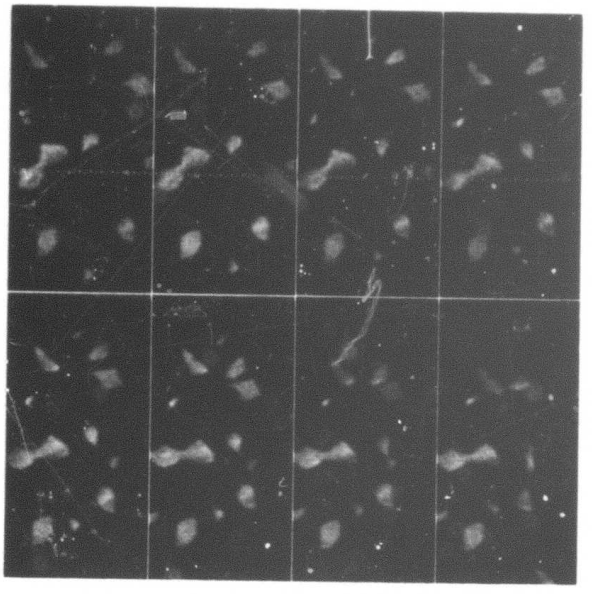
Figure 2 shows some frames from two such sequences with different backgrounds.

In order to carry out the motion and shape analysis on some real sequences of images, we have also carried out experiments at the USC medical school, to analyze the behavior of lymphocytes in the presence of cancer cells in vitro. The TV camera in Fig. 1 is connected to a NIKON phase contrast microscope. A confluent monolayer of human skin cancer cells was grown on microslides. Using this slide a Sykes-Moore chamber was formed and filled with a RPMI 1640 media with 10% serum. Lymphocytes were isolated from fresh human blood by gelatine technique and their concentration was about 90%. A small number of these lymphocytes were injected into the chamber. The whole system was incubated at 37^{o}c. The movement of the lymphocytes was recorded at 30 sec. to 60 sec. intervals in the images (128x128 pixels) over a period of about 24 hours. 128x128 pixel image corresponds to an area of 60 μm x 60 μm of the slide at x200 magnification. Some representative pictures are shown in Fig. 3.

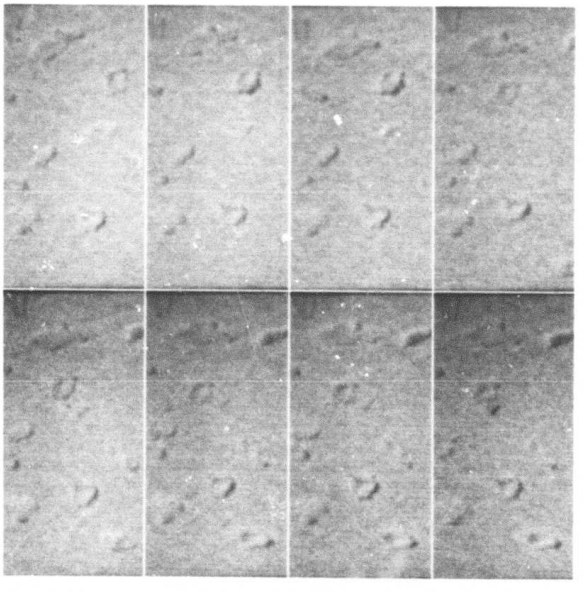
Dynamic Image Understanding Model

For the motion and shape analysis of objects in an adaptive manner we consider a new dynamic image understanding model presently under investigation. Its simplified block diagram is shown in Fig. 4.

As seen from Fig. 4 that this model achieves the symbols, features and segmentation control by distributed feedback. It is similar to the heterarchical model suggested by Minsky & Papert and used by Shirai [10] for obtaining line drawings. On the basis of the fact that the predicted model of the scene at a particular time matches with the scene at that time, we do simple or complicated



(a)



(b)

Fig. 2. Typical input sequences with different backgrounds.

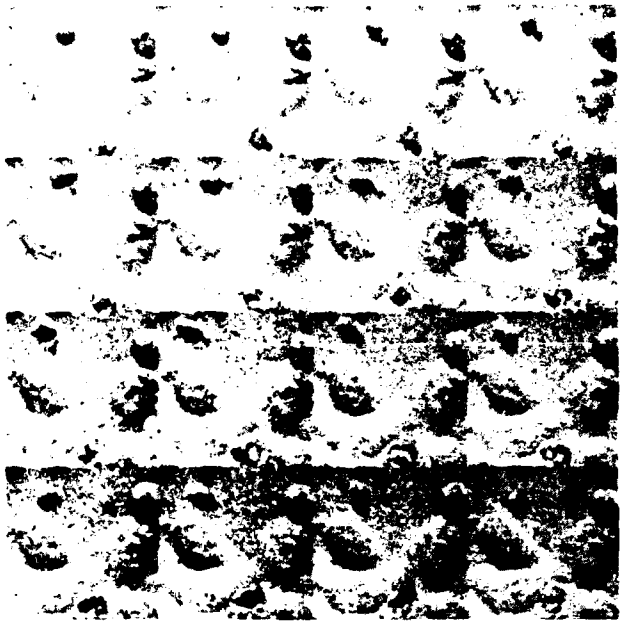


Fig. 3. A real sequence of 16 images.

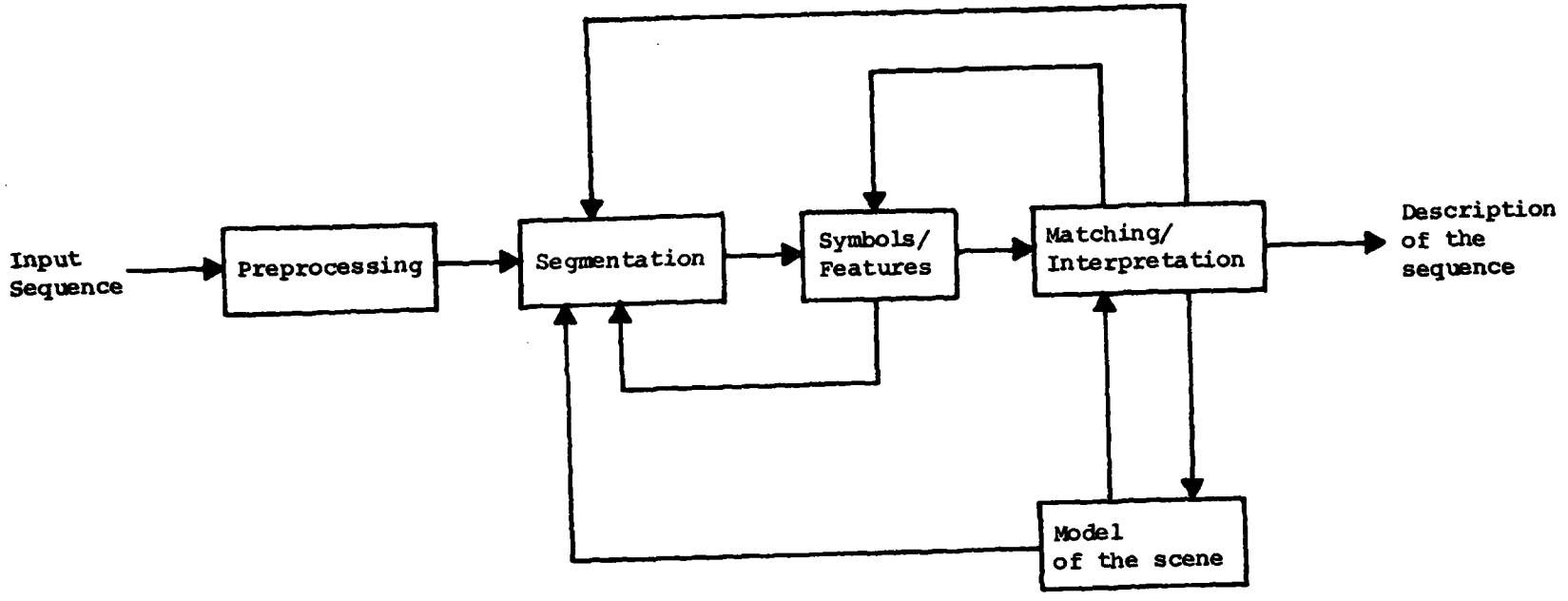


Figure 4. System for the analysis of image sequences

processing for motion as well as shape analysis. Also segmentation varies with the dyramics of the scene.

Throughout this study emphasis will be given on the iterative, parallel processes, known as "relaxation" techniques [11-13]. In general, "relaxation" is an iterative approach for classifying a set of interrelated objects. In "discrete relaxation" a set of possible class names is initially associated with each object. At subsequent iterations, class names are discarded from an object if they are inconsistent with the surviving class name possibilities for other, related objects; this is done for all objects simultaneously. The process is repeated until no further changes can take place. In "probabilistic relaxation", a set of estimates of class assignment probabilities is initially associated with each object. At subsequent iterations, the probabilities are adjusted in accordance with the support they receive from the class probabilities of related objects. This process, when repeated often leads to a marked reduction in the ambiguity of classification.

Ullman [10] and Faugeras and Berthod [62] consider the relaxation processes as problems of computing constrained optimization by local processes. The algorithm of Rosenfeld et. al.[12] uses no explicit measure of ambiguity of classification. Faugeras and Berthod formulate the relaxation algorithm in terms of minimizing a criterion function which includes the inconsistency and ambiguity of classification.

Barnard and Thompson [14] and Ullman [10] consider the application of relaxation algorithms in the matching of two images. In [14] an initial network of possible matches between the two sets of candidates in two images is constructed. Each possible match specifies a possible disparity of a candidate point in a related reference image. An initial estimate of the probability of each possible disparity is made, based on the similarity of subimages surrounding the points. These estimates are iteratively improved by

using the relaxation algorithm similar to of Rosenfeld et. al. [12].

There are a number of drawbacks in merely using the matching approach for analyzing the motion of objects in a sequence of images. First, although such matching approaches incorporate spatial correlation within a image, they do not consider temporal correlation at all. Since the successive images are highly correlated, the inclusion of this information should be helpful in "better" and "fast" matching. Second, if such a matching approach is applied to a large number of frames, it will be very expensive computationally.

Study removing such drawbacks and the details of the model along with the computational issues related to analyzing the sequences of images are being worked out and will be reported in the future.

Acknowledgements

The authors wish to thank Professor D. Marsh of USC Medical School for providing the real time video acquisition and digital display system, and Professor Z. Toke's of USC Cancer Research Center for providing the cell cultures. Help of Mr. Csaba relating to cell cultures is also appreciated.

References

1. Workshop on Computer Analysis of Time Varying Imagery, April 5-6, 1979, Philadelphia, Pa.

2. Martin, W.N., and J.K. Aggrawal, "Dynamic Scene Analysis," CGIP, Vol. 7, 1978, pp. 356-374.

3. Nagel, H.H., "Analysis Techniques for Image Sequences," Proc. IJCPR4, Nov. 1978, pp. 186-211.

4. Yachida, M., M. Asada, and S. Tsuji, "Automatic Motion Analysis

System of Moving Objects from the Records of Natural Processes," Proc. IJCPR4, Nov. 1978, pp. 726-730.

5. Bhanu, B., "Bibliography on the Tracking of objects moving in space," May 9, 1979, Image Processing Institute, USC, Los Angeles, Ca.

6. Mayeda, T., "Real Time Video Acquisition and Digital Display System," Instructional Manual, Image Processing Institute, USC, Los Angeles, Ca.

7. Bhanu, B., "On the analysis techniques for image sequences," Unpublished Ph.D. dissertation proposal, Image Processing Institute, USC, Oct.-Nov. 1979 (Advisor:Prof. O.D. Faugeras).

8. Aggarwal, J.K. and R.O. Duda, "Computer Analysis of Moving Polygonal Images," IEEE Trans. on Computers, Vol. C-24, Oct. 1975, pp. 966-976.

9. Martin, W.N. and J.K. Aggarwal, "Computer Analysis of Dynamic Scenes containing curvilinear figures," Pattern Recognition, Vol. 11, 1979, pp. 169-178.

10. P.H. Winston, Ed., "The psychology of computer vision," McGraw Hill, 1975.

11. Faugeras, O.D., and M. Berthod, "Scene Labeling: An Optimization Approach," Proc. IEEE Comp. Soc. Conf. on PRIP, Aug. 6-8, 1979, pp. 318-326.

12. Rosenfeld A., R. Hummel and S.W. Zucker, "Scene labeling by relaxation operations," IEEE Trans. on SMC, Vol. 6, 1976, pp. 420-433.

13. Ullman, S., "Relaxation and constrained optimization by local processes," CGIP, Vol. 10, 1979, pp. 115-125.

14. S.T. Barnard and W.B. Thompson, "Disparity analysis of images," Tech. Report 79-1, Dept. of Comp. Sci., Univ. of Minnesota.

2.10 Segmentation of Images having Unimodal Gray Level Distributions

B. Bhanu and O.D. Faugeras

Introduction

Image segmentation is a transformation from a point by point representation of an image to a representation of the image as a collection of meaningful regions. In image analysis it is the most primitive process, because a segmented image requires less storage than the original and can be analyzed on the basis of shapes, features and other characteristics of its regions. Various approaches based on thresholding have been used by many researchers for the segmentation of both monochrome and color pictures [1-4]. A good survey of global, local and dynamic threshold selection methods is given in [1].

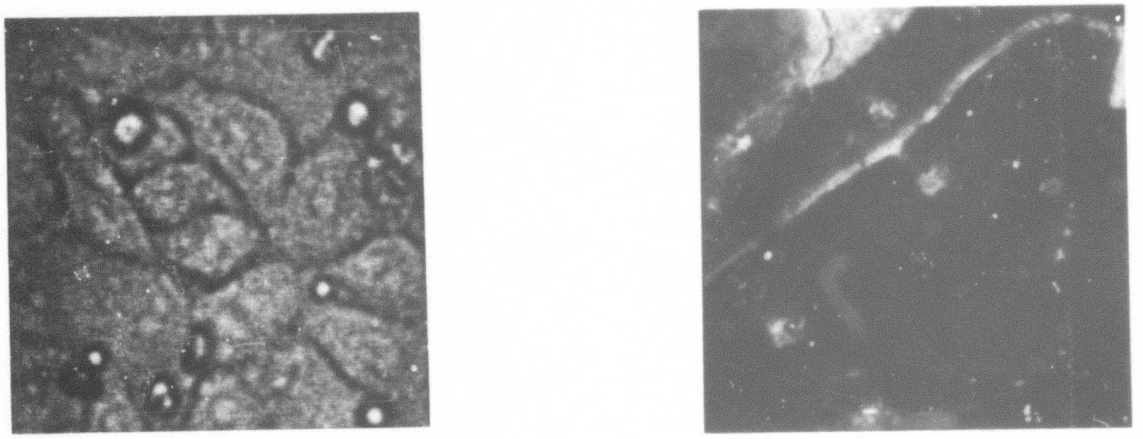
Normally, in the application of these techniques, the gray level histogram of the image shows two or more peaks and the preprocessing is done to improve the histogram, if needed. Local properties are also used to compute the global, local or dynamic thresholds. However, if the gray level histogram of the image is unimodal, then the application of such methods give poor segmentation results and there is no criteria for automatic threshold selections. In [8,9] Rosenfeld and Davis discuss the histogram modification iteratively. Various pictures shown in [8] already consist of 2 or more peaks. In this paper we consider the application of relaxation methods for segmenting monochrome pictures having unimodal gray level

distributions. It is found that the gradient relaxation method gives very good segmentation and provides the automatic selection of thresholds. Results are illustrated with the aid of examples.

Segmentation Schemes

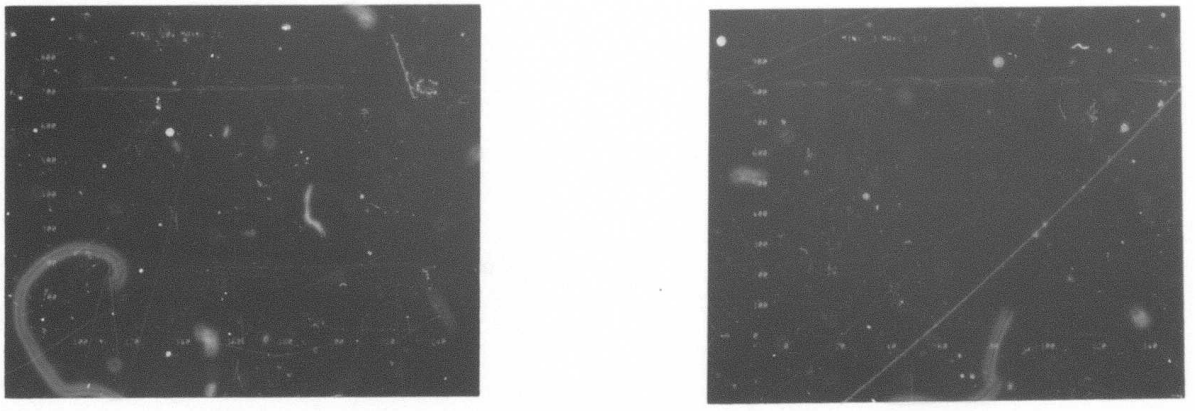
Figure 1 shows two 128x128 pixels 8-bit images. Image in fig. 1(a) has been obtained using the experimental setup discussed in another paper on "Computer Analysis of Moving Images" of this report. The background in this image consists of a confluent monolayer of human skin cancer cells and the small circular shaped objects are human lymphocytes and red blood cells. The objective is to get the boundaries of all the cells. Image in fig. 1(b) is taken from DMA3.2048. Here the objective is to detect roads etc. Gray level histogram of these images are shown in Fig. 2. Note that the histograms are Gaussian shaped. Since the histograms are unimodal, there is no automatic method for segmenting these pictures.

Commonly used difference operations such as gradient, Laplacian and Sobel were applied on the images shown in Fig. 1. We also considered the methods based on thresholding the histogram of the picture where the gradient, Laplacian and edge values are high [1]. However, the picture so obtained has unimodal histogram and lacks the criterion for segmenting it at the valley of two peaks. Thresholding at the gray level corresponding to the mode or mean of the filtered histogram gave very poor results. Edge detection has also been done by convolving the image in Fig. 1 with 5x5 masks corresponding to the ideal step edges in six directions as described in [5]. The magnitude of the convolved output and the direction of the mask giving the highest output at each pixel are recorded as edge data. The magnitude image does not show good segmentation. A number of bar masks of various sizes and orientations have also been used, but the results were poor (for the image shown in Fig. 1a the width of an edge is about 5-6 pixels).



(a) (b)

Fig. 1. Two typical 128x128, 8 bit images.
(a) Cell image, (b) Image taken from DMA3.2048



(a) (b)

Fig. 2. Gray level histogram of the images in Fig. 1.
(a) Histogram of the image in fig. 1(a). Mean=163.76,
(b) Histogram of the image in fig. 1(b). Mean=26.47.

Segmentation Using Relaxation Methods

Having considered the basic differencing, edge, bar operators and their variations, we focussed our attention on the probabilistic relaxation methods for segmentation [6,7]. "Relaxation" is an iterative approach to classifying a set of interrelated objects. In "probabilistic relaxation," a set of estimates of class assignment probabilities is initially associated with each object. At subsequent iterations, the probabilities are adjusted in accordance with the support they receive from the class probabilities of related objects. This process, when repeated often leads to a marked reduction in the ambiguity of classification.

Nonlinear Probabilistic Relaxation Algorithm

First we consider the nonlinear probabilistic relaxation algorithm proposed in [6] which has been extensively used in edge detection, edge and line enhancement, clustering etc. In order to use this algorithm for segmentation purposes, suppose we have a set of N objects (pixels) $a_1,\ldots,a_N$, which fall into two classes λ_1 and λ_2 corresponding to the white, and black gray values. For the same neighborhood V_i of the two classes (known as homogeneous case), the probability of being white at the (n+1)th iteration is given by,

$$p_i^{(n+1)}(\lambda_k) = \frac{p_i^{(n)}(\lambda_k) q_i^{(n)}(\lambda_k)}{\sum_{\ell=1}^{2} p_i^{(n)}(\lambda_\ell) q_i^{(n)}(\lambda_\ell)} \qquad \begin{matrix} k = 1,2 \\ i = 1,\ldots,N \end{matrix} \qquad (1)$$

where the consistency vector $\vec{q}_i$ is,

$$q_i(\lambda_k) = \sum_{j \in V_i} \alpha_{ij} \sum_{\ell=1}^{2} p_{ij}(\lambda_k|\lambda_\ell) p_j(\lambda_\ell) \qquad i = 1,\ldots,N \quad (2)$$

and

$$\sum_{j \in V_i} \alpha_{ij} = 1$$

α_{ij}'s are called as compatibility coefficients. Considering the neighborhood V_i of 8 around the central pixel a_i for both classes and taking equal values of α_{ij}'s Eq. (2) becomes,

$$q_i(\lambda_k) = \frac{1}{8} \sum_{j \in V_i} \sum_{\ell=1}^{2} p_{ij}(\lambda_k|\lambda_\ell) p_j(\lambda_\ell) \quad (3)$$

$p_{ij}(\lambda_k|\lambda_\ell)$ is the conditional probability that object a_i belongs to class λ_k given that object $a_j \in V_i$ belongs to class λ_ℓ. For the two class problem, we take

$$p_{ij}(\lambda_1|\lambda_1) = p_{ij}(\lambda_2|\lambda_2) = 1 \quad (4)$$

and

$$p_{ij}(\lambda_1|\lambda_2) = p_{ij}(\lambda_2|\lambda_1) = 0 \quad (5)$$

Thus,

$$q_i(\lambda_1) = \frac{1}{8} \sum_{j \in V_i} p_j(\lambda_1) \quad (6)$$

Noting that

$$p_i(\lambda_2) = 1-p_i(\lambda_1)$$

and

$$q_i(\lambda_2) = 1-q_i(\lambda_1)$$

Eq. (1) can be written as,

$$p_i^{(n+1)}(\lambda_1) = \frac{p_i^{(n)}(\lambda_1)q_i^{(n)}(\lambda_1)}{1+2p_i^{(n)}(\lambda_1)q_i^{(n)}(\lambda_1)-p_i^{(n)}(\lambda_1)-q_i^{(n)}(\lambda_1)} \tag{7}$$

So at each iteration given the values of $p_i^{(m)}(\lambda_1)$, we compute $q_i^{(m)}(\lambda_1)$ from Eq. (6) and then $p_i^{(m+1)}(\lambda_1)$ from Eq. (7).

In the case when we consider the neighborhood of 8 for class λ_1 and the neighborhood 4 for class λ_1, Eq. (6) becomes,

$$q_i(\lambda_1) = \frac{\frac{1}{8}\sum_{j\in V_i(\lambda_1)} p_j(\lambda_1)}{\frac{1}{8}\sum_{j\in V_i(\lambda_1)} p_j(\lambda_1)+\frac{1}{4}\sum_{j\in V_i(\lambda_2)} p_j(\lambda_2)}$$

or

$$q_i(\lambda_1) = \frac{\frac{1}{8}\sum_{j\varepsilon V_i} p_j(\lambda_1)}{\frac{1}{8}\sum_{j\varepsilon V_i(\lambda_1)} p_j(\lambda_1) + 1 - \frac{1}{4}\sum_{j\varepsilon V_i(\lambda_2)} p_j(\lambda_1)} \tag{8}$$

The initial assignment of probabilities $p_i^{(0)}(\lambda_1)$ can be done by simply normalizing the gray values of the image by 255. However, it does not take into account the mean value of the picture. So the following transformation is used,

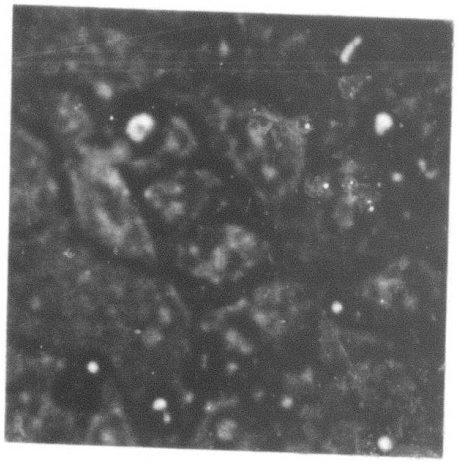
$$\text{FACT} * \left(\frac{\text{I-Global mean}}{255}\right) + 0.5 \tag{9}$$

where I = intensity value at a pixel and FACT is a constant whose value is determined as follows:

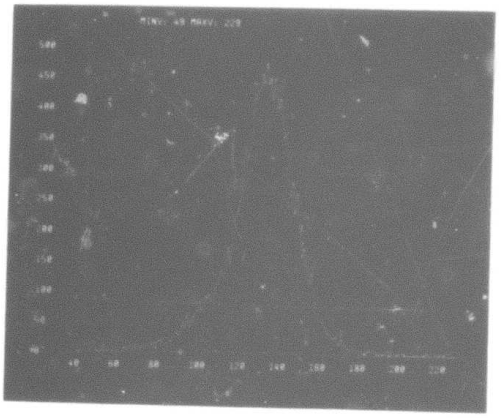
$$\begin{aligned} \text{Fact} &= 1 && \text{if I} > \text{Global mean} \\ &= 0.7 \text{ to } 0.9 && \text{if I} < \text{Global mean} \end{aligned} \tag{10}$$

In Eq. (9) if $\text{Fact}*\left(\frac{\text{I-Global mean}}{255}\right) < -0.5$, then $p_i^{(0)}(\lambda_1)$ is taken as zero at that pixel.

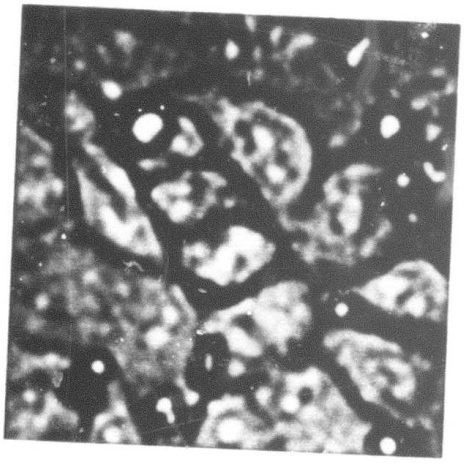
This transformation is motivated by the fact that the gray level histogram of a typical natural image is usually skewed toward the darker gray values and therefore, the majority of the pixels possess gray value less than average. Figs. 3 and 4 illustrate the application of this method to the images shown in Fig. 1. As can be seen from the histogram plots, in the first few iterations a sort of histogram equalization takes place and then black and white classes



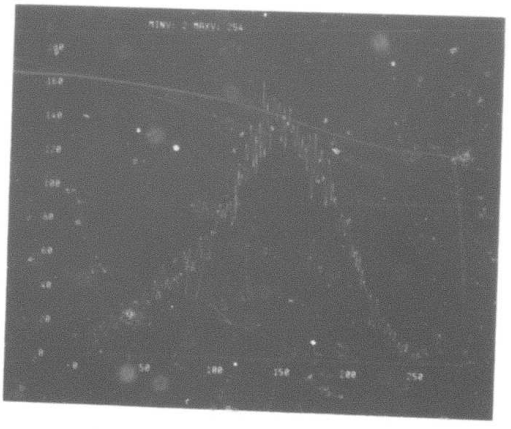
(a) Iteration 1



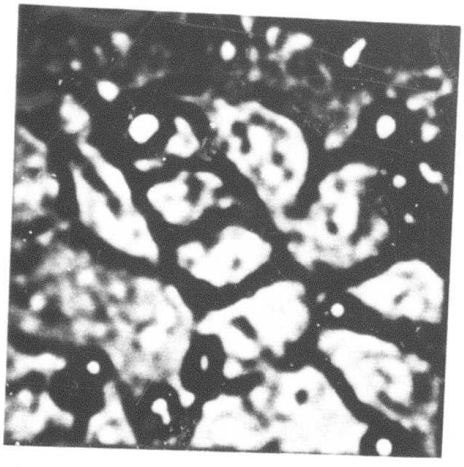
(b) Histogram of fig. 3(a)



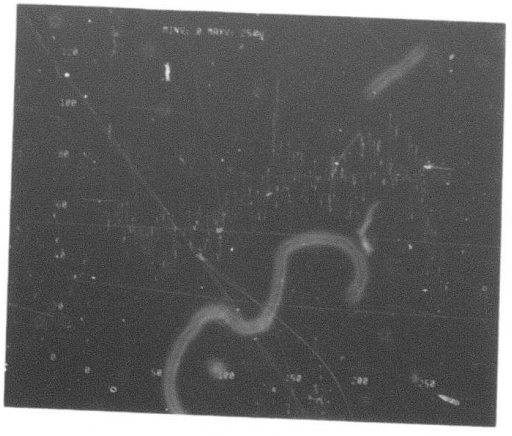
(c) Iteration 3



(d) Histogram of fig. 3(c)

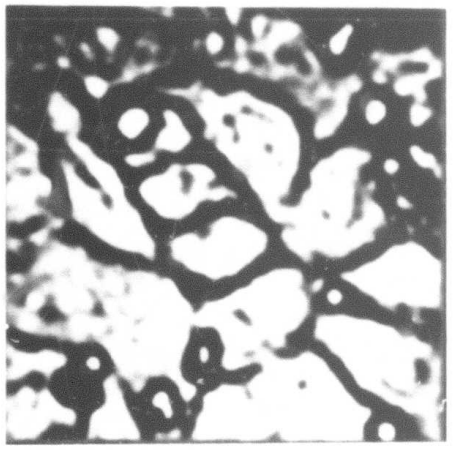


(e) Iteration 4

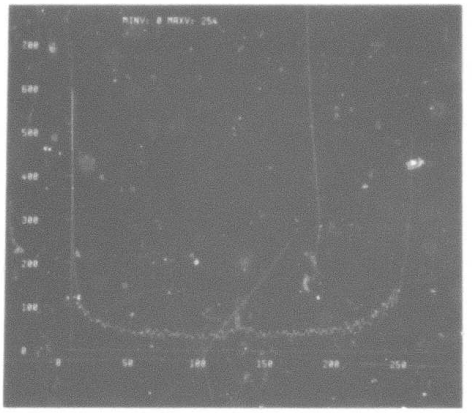


(f) Histogram of fig. 3(e)

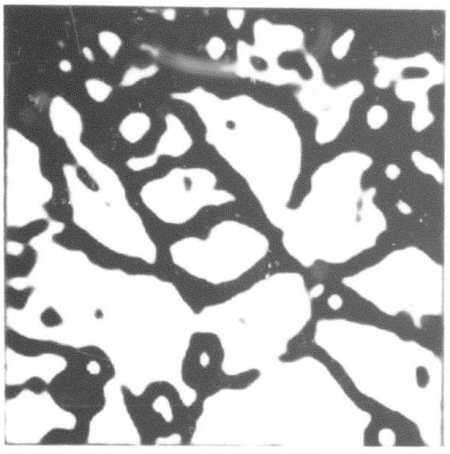
Fig. 3. Results of Nonlinear relaxation method at various iterations and corresponding histograms for the image in fig. 1(a). FACT=0.9 (see text).



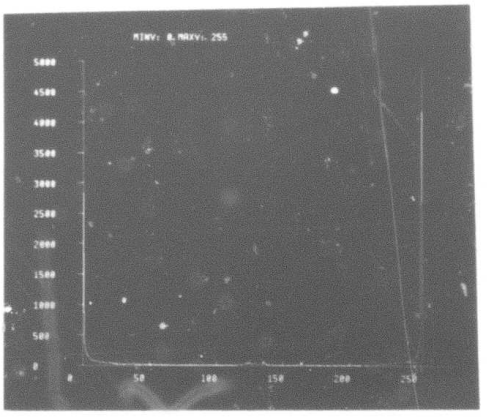
(g) Iteration 5



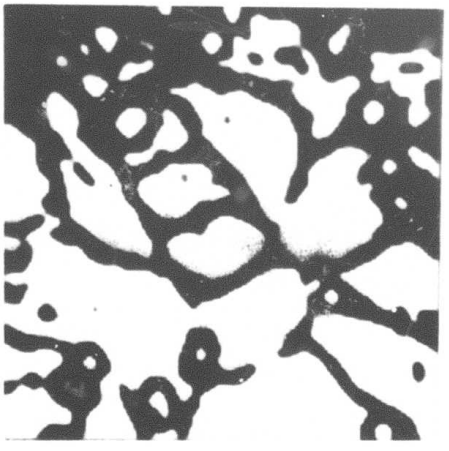
(h) Histogram of fig. 3(g)



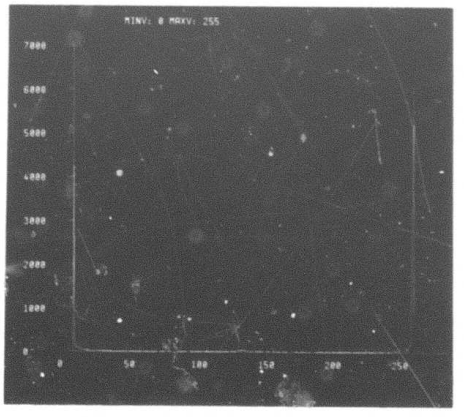
(i) Iteration 7



(j) Histogram of fig. 3(i)

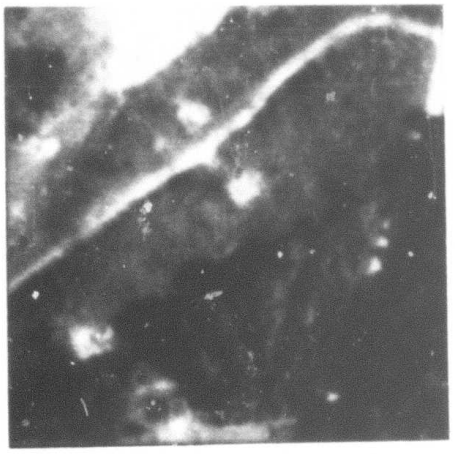


(k) Iteration 9

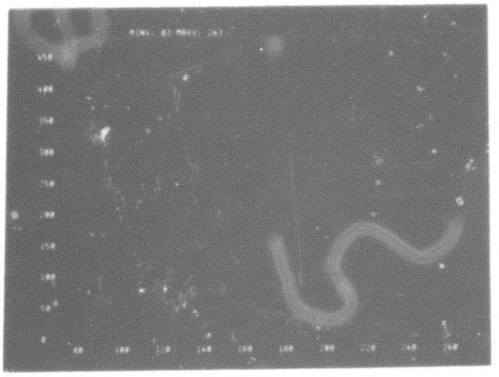


(l) Histogram of fig. 3(k)

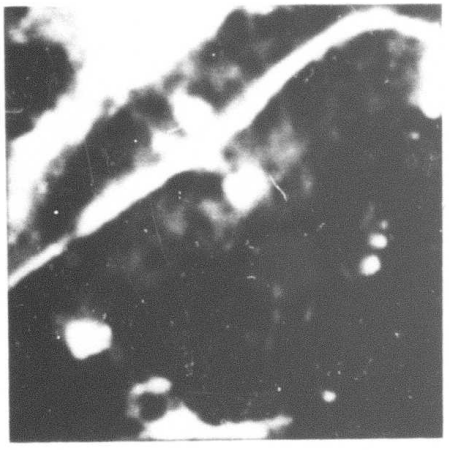
Fig. 3. (CONT.)



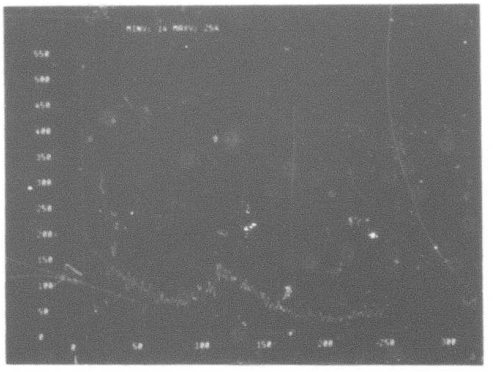
(a) Iteration 1



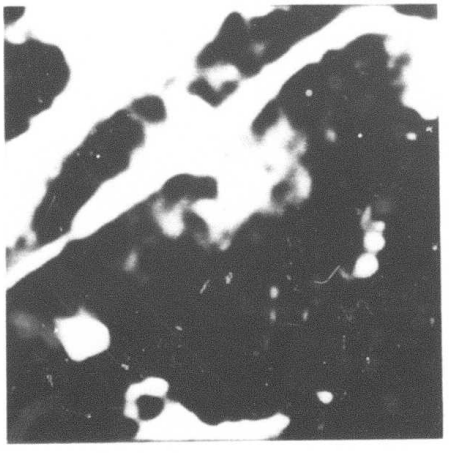
(b) Histogram of fig. 4(a)



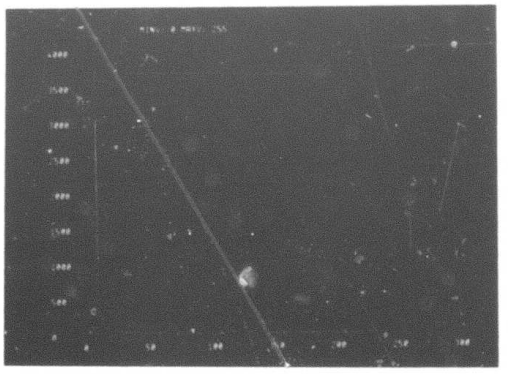
(c) Iteration 3



(d) Histogram of fig. 4(c)



(e) Iteration 5



(f) Histogram of fig. 4(e)

Fig. 4. Results of Nonlinear relaxation method at various iterations and corresponding histograms for the image in fig. 1(b). FACT=1 (see text).

get separated. However, in this method convergence is slow, uncontrolled and does not lead to the automatic selections of threshold at the valley of two peaks which are not far apart. Results obtained by thresholding at the mean are discussed in the next section. Gradient relaxation method discussed next overcomes these disadvantags.

Gradient Relaxation Algorithm

Faugeras and Berthod [7] proposed a different approaches for relaxation algorithm which is based upon the explicit use of consistency and ambiguity to define a global criterion upon the set of objects. The criterion is minimized using the projection gradient techniques. In [7] they consider the minimization of the criterion,

$$C_3 = \alpha C_1 + (1-\alpha) C_2 \tag{11}$$

where, α varies between 0 and 1, and weights the relative importance we assess to consistency versus ambiguity, and

$$C_1 = \frac{1}{2N} \sum_{i=1}^{N} \| \vec{p}_i - \vec{q}_i \|_2^2 \tag{12}$$

$$C_2 = \frac{L}{L-1} \left[1 - \frac{1}{N} \sum_{i=1}^{N} \| \vec{p}_i \|_2^2 \right] \tag{13}$$

However, in the present study we consider a simpler criterion based on the dot product of $\vec{p}_i$ and $\vec{q}_i$,

$$C = \sum_{i=1}^{N} \vec{p}_i \cdot \vec{q}_i \tag{14}$$

and carry out its maximization using the projection gradient approach. This criterion is easier to manipulate computationally.

For the two class case λ_1, λ_2 and the same 8-neighborhood for these classes and assuming the same conditional probabilities as in the earlier discussion on nonlinear relaxation method, gradient of the criterion C is obtained as,

$$\frac{\partial C}{\partial p_i(\lambda_1)} = 2q_i(\lambda_1) \tag{15}$$

$$\frac{\partial C}{\partial p_i(\lambda_2)} = 2q_i(\lambda_2) \tag{16}$$

The initial assignment of probabilities is taken to be the same as in the nonlinear relaxation method. The iteration of p_i's is given by,

$$p_i^{(n+1)}(\lambda_1) = p_i^{(n)}(\lambda_1) + \rho_i p_i^{(n)} \left[\frac{\partial C}{\partial p_i(\lambda_1)}\right] \tag{17}$$

$$p_i^{(n+1)}(\lambda_2) = p_i^{(n)}(\lambda_2)+\rho_i p_i^{(n)}\left[\frac{\partial C}{\partial p_i(\lambda_2)}\right] \tag{18}$$

In order to have $p_i(\lambda_1)+p_i(\lambda_2) = 1$, the projection of the gradient should be such that [7],

$$p_i^{(n)}\left[\frac{\partial C}{\partial p_i(\lambda_1)}\right] = 2q_i(\lambda_1)-\frac{1}{2}\left[\frac{\partial C}{\partial p_i(\lambda_1)}+\frac{\partial C}{\partial p_i(\lambda_2)}\right] \tag{19}$$

and

$$p_i^{(n)}\left[\frac{\partial C}{\partial p_i(\lambda_2)}\right] = 2q_i(\lambda_2)-\frac{1}{2}\left[\frac{\partial C}{\partial p_i(\lambda_1)}+\frac{\partial C}{\partial p_i(\lambda_2)}\right] \tag{20}$$

but

$$\frac{\partial C}{\partial p_i(\lambda_1)} + \frac{\partial C}{\partial p_i(\lambda_2)} = 2$$

So the iteration eqs. in (17) and (18) reduce to

$$p_i^{(n+1)}(\lambda_1) = p_i^{(n)}(\lambda_1)+\rho_i^{(n)}[2q_i(\lambda_1)-1] \tag{21}$$

$$p_i^{(n+1)}(\lambda_2) = p_i^{(n)}(\lambda_2)+\rho_i^{(n)}[2q_i(\lambda_2)-1] \tag{22}$$

Normally, $p_i^{(n)}$ is kept constant for all pixels during each iteration and is determined to have the largest possible value such that p_i's at (n+1)th iteration still lie in the bounded convex region of 2N dimensional Euclidean space. However, in the 2 class considered it is easier to compute $p_i^{(n)}$ for each pixel (although we may not be

following the gradient exactly) and is obtained from Eq. (21) and (22) as,

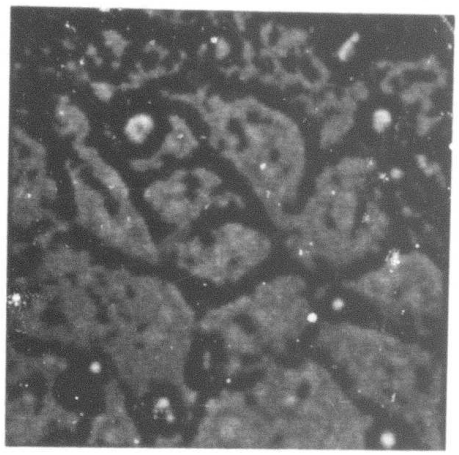
$$\rho_i^{(n)} = \alpha_1 \left(\frac{1-p_i^{(n)}(\lambda_k)}{2q_i(\lambda_k)-1} \right), \qquad \text{if } 2q_i(\lambda_k)-1 > 0 \tag{23}$$

$$= \alpha_2 \left(\frac{p_i^{(n)}(\lambda_k)}{1-2q_i(\lambda_k)} \right), \qquad \text{if } 2q_i(\lambda_k)-1 < 0 \tag{24}$$

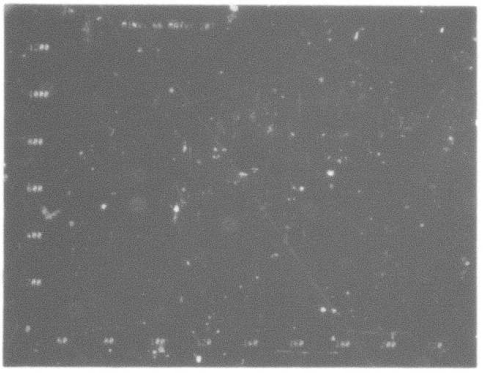
where K=1,2, and α_1 and α_2 are constants less than 1. These values may be selected to bias a class and control the speed of convergence.

Figs. 5 and 6 show results obtained at various iterations and corresponding histograms. Note that at the first iteration itself we get two peaks separated by a valley which can be used to automatically select the threshold to obtain segmentation. As the number of iteration increases the two peaks get apart, average brightness increases, and the convergence of probabilities takes place as expected. When the peaks are far apart, thresholding can be done at the mean value. Segmentation result obtained by semi-thresholding the gray level histogram are shown in Figs. 7 and 8. These clearly show that gradient relaxation method gives better segmentation than nonlinear relaxation. For the nonlinear relaxation case, we have shown the segmentation results only when the two peaks are separated.

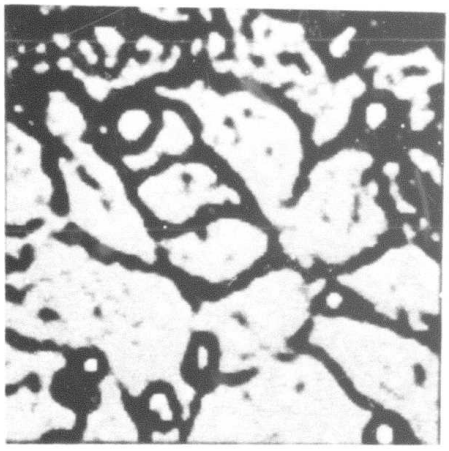
In the above example we have considered the 8-neighborhood for both classes λ_1 and λ_2. If we consider the 8-neighborhood for class λ_1 and 4-neighborhood for class λ_2, the gradients are obtained as.



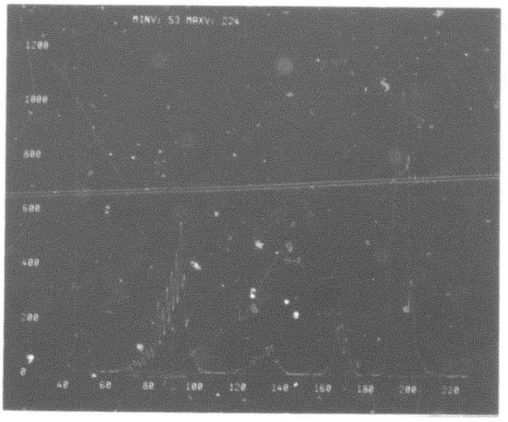
(a) Iteration 1



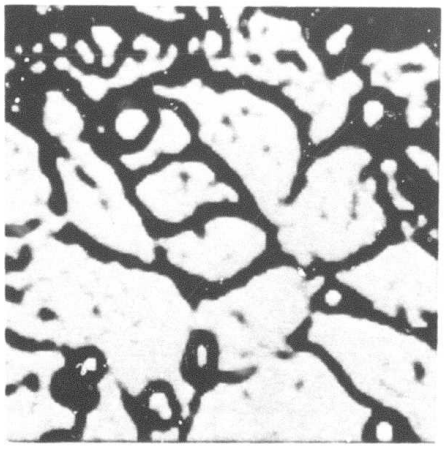
(b) Histogram of fig. 5(a)



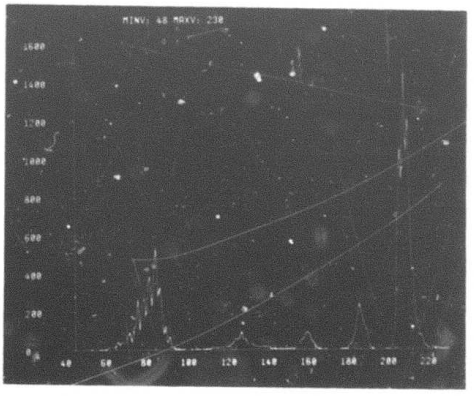
(c) Iteration 3



(d) Histogram of fig. 5(c)

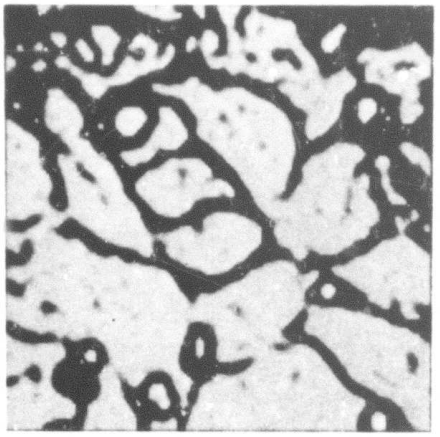


(e) Iteration 4

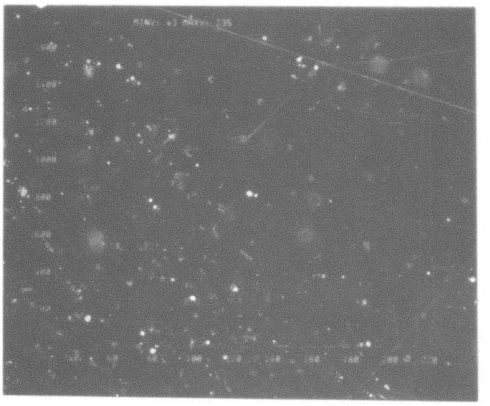


(f) Histogram of fig. 5(e)

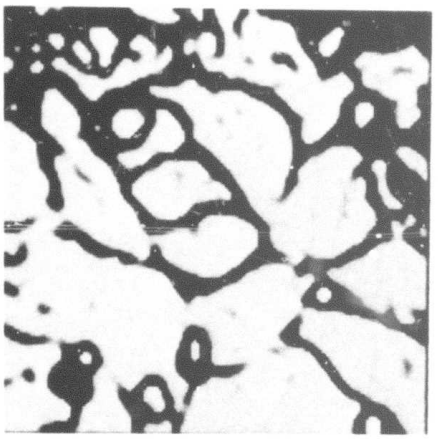
Fig. 5. Results of Gradient Relaxation method at various iterations and corresponding histograms for the image in fig. 1(a). FACT=0.9, α_1=0.2, α_2=0.1 (see text).



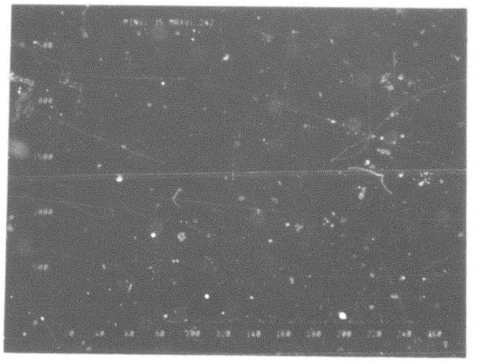
(g) Iteration 5



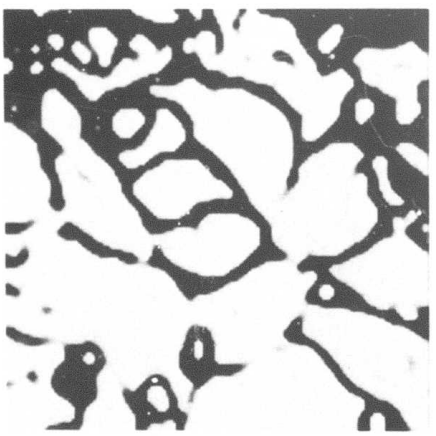
(h) Histogram of fig. 5(g)



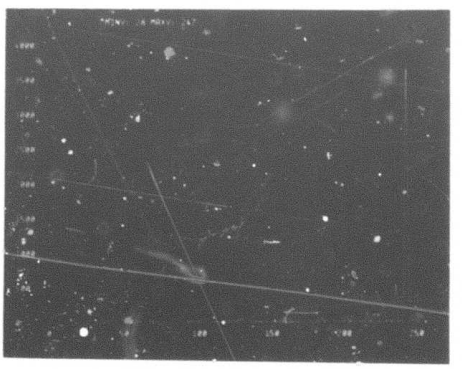
(i) Iteration 7



(j) Histogram of fig. 5(i)

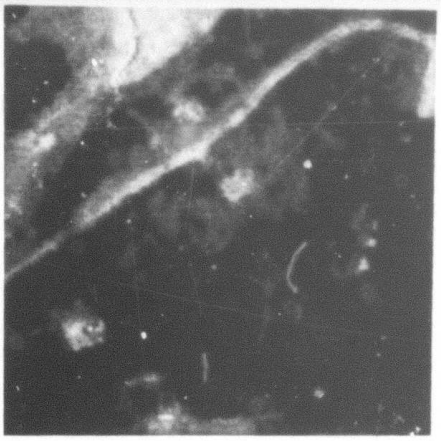


(k) Iteration 9

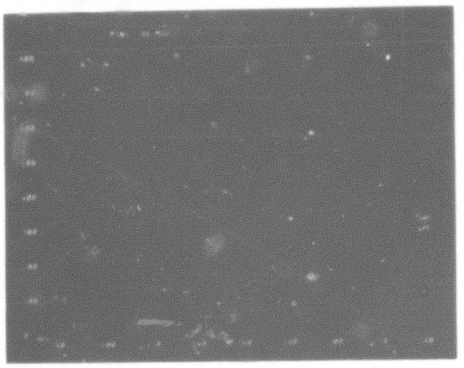


(l) Histogram of fig. 5(k)

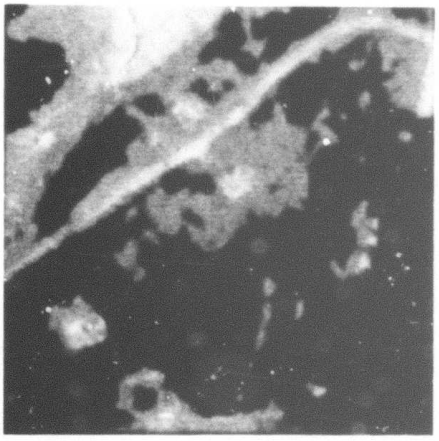
Fig. 5. (CONT.)



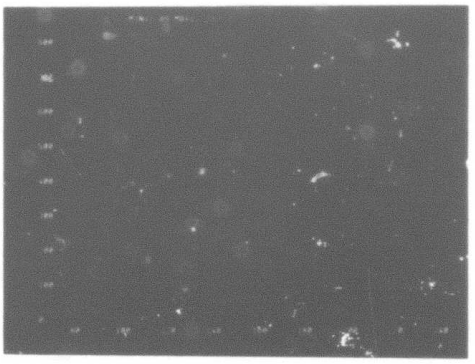
(a) Iteration 1



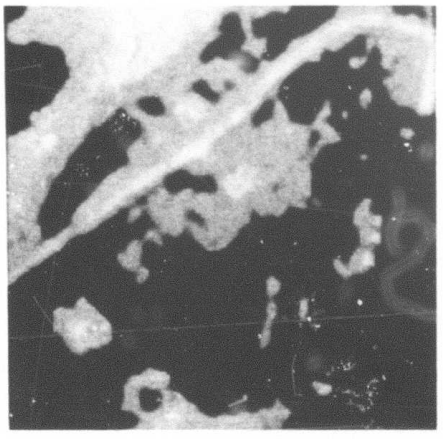
(b) Histogram of fig. 6(a)



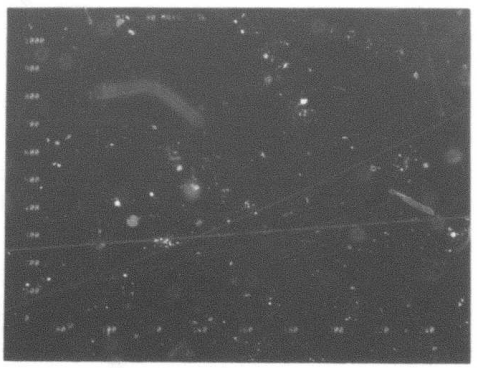
(c) Iteration 3



(d) Histogram of fig. 6(c)

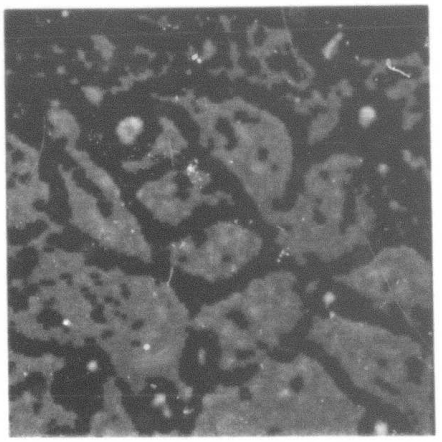


(e) Iteration 5



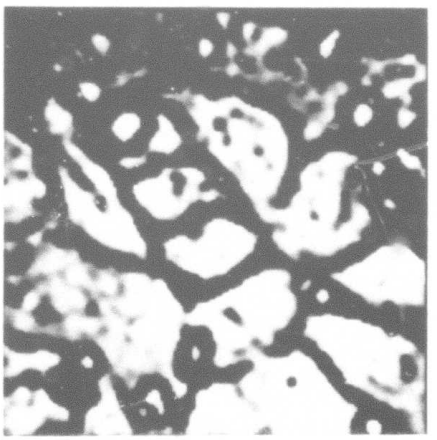
(f) Histogram of fig. 6(e)

Fig. 6. Results of Gradient Relaxation method at various iterations and corresponding histograms for the image in fig. 1(b). FACT=1, α_1=0.1, α_2=0.05 (see text).



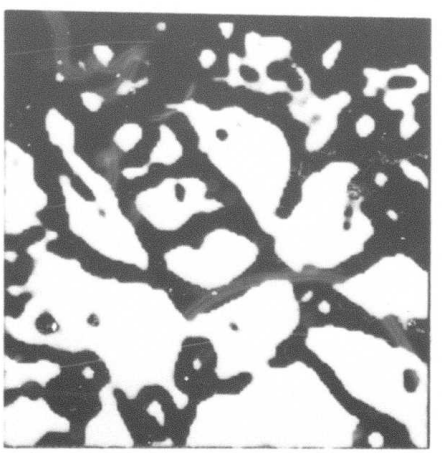
(a) Semi-thresholding of fig. 5(a) at the valley bottom (see fig. 5(b))

(b) Semi-thresholding of fig. 5(g) at the mean



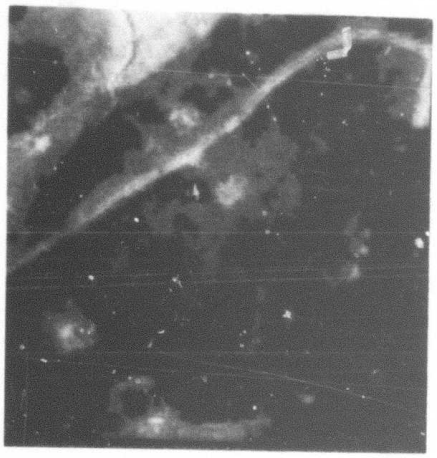
(c) Semi-thresholding of fig. 3(g) at the mean

(d) Semi-thresholding of fig. 5(i) at the mean

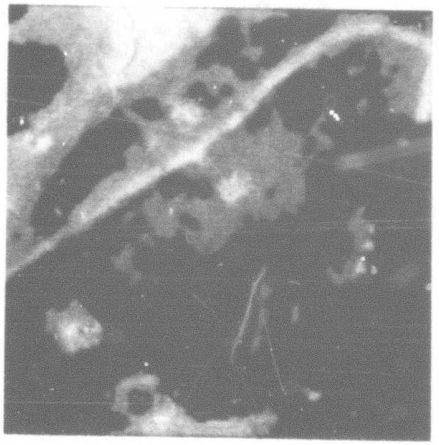


(e) Semi-thresholding of fig. 3(i) at the mean

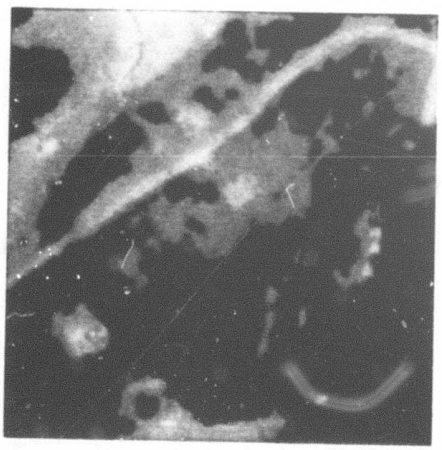
Fig. 7. Segmentation results for Gradient and Nonlinear relaxation methods obtained by semi-thresholding at various iteration for the image in fig. 1(a).



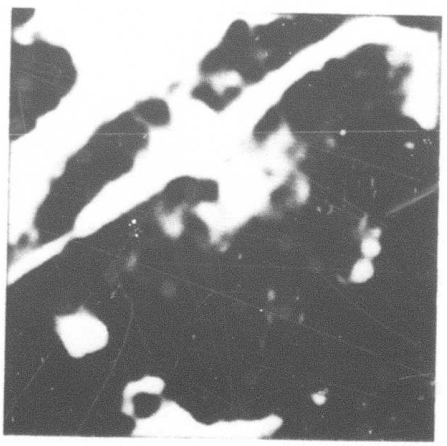
(a) Semi-thresholding of fig. 6(a) at the valley bottom (see fig. 6(b))



(b) Semi-thresholding of fig. 6(c) at the mean



(c) Semi-thresholding of fig. 6(e) at the mean



(d) Semi-thresholding of fig. 4(e) at the mean

Fig. 8. Segmentation results for Gradient and Nonlinear relaxation methods obtained by semi-thresholding at various iterations for the image in fig. 1(b).

$$\frac{\partial C}{\partial p_i(\lambda_1)} = q_i(\lambda_1) + \sum_{j \varepsilon V_i(\lambda_1)} \frac{1}{8D_j} [p_j(\lambda_j) - \vec{p}_j \cdot \vec{q}_j] \tag{25}$$

and

$$\frac{\partial C}{\partial p_i(\lambda_2)} = q_i(\lambda_2) + \sum_{j \varepsilon V_i(\lambda_2)} \frac{1}{8D_j} [p_j(\lambda_2) - \vec{p}_j \cdot \vec{q}_j] \tag{26}$$

where

$$q_i(\lambda_1) = \frac{\frac{1}{8} \sum_{j \varepsilon V_i(\lambda_1)} p_j(\lambda_1)}{D_j} \tag{27}$$

$$q_i(\lambda_2) = \frac{\frac{1}{4} \sum_{j \varepsilon V_i(\lambda_2)} p_j(\lambda_2)}{D_j} \tag{28}$$

and

$$D_j = \frac{1}{8} \sum_{j \varepsilon V_i(\lambda_1)} p_j(\lambda_1) + \frac{1}{4} \sum_{j \varepsilon V_i(\lambda_2)} p_j(\lambda_2) \tag{29}$$

Now the projection of the gradient can be obtained as before. The application of the iteration thus obtained gave the results very

much like those obtained using Eq. (21).

This method has been applied on a number of aerial and other pictures having unimodal distributions and gave good segmentation results.

Conclusions

It has been shown that the gradient method of relaxation can be used as a tool for segmentation of black and white pictures having unimodal distributions. This method provides automatic selection of the threshold which leads to good segmentation results.

In this study we have considered only two classes. In scenes with many different objects as in the case with aerial photographs, there are more classes although the histogram may have only one peak because the range of intensities for each object will probably overlap with the ranges of other objects. Two generalizations are possible. In the first we simply extend the method for more classes, which are to be known a priori. In the second case, we apply the procedure outlined in this paper to each segmented region iteratively in a tree structure until the area of a region has reduced to a specified limit. If a region is homogeneous in intensity, then we will not expect to find two peaks and will stop iterating after a predefined number of iterations. An experiment investigating the sensitivity of this method to homogeneous region with varying amount of noise will be interesting.

References

1. J.S. Weszka, "A Survey of Threshold Selection Techniques," CGIP 7, pp. 259-265, 1978.

2. K.E. Price, "Change Detection and Analysis in Multi-Spectral Images," Ph.D. Thesis 1976, Dept. of Comp. Sci., Carnegie-Mellon

University.

3. A. Rosenfeld and A.C. Kak, "Digital Picture Processing," Academic Press, New York, 1976.

4. T. Pavlidis, "Structural Pattern Recognition," Springer-Verlag, 1977.

5. R. Nevatia and K.R. Babu, "An Edge Detection, Linking and Line Finding Program," USCIPI Report 840, pp. 103-116, Sept. 1978.

6. A. Rosenfeld, R. Hummel and S.W. Zucker, "Scene Labeling by Relaxation Operations," IEEE Trans. on SMC, Vol. 6, pp. 420-433, 1976.

7. O. Faugeras and M. Berthod, "Scene Labeling: An Optimization Approach," Proc. IEEE Comp. Soc. Conf. on PRIP, Aug. 6-8, 1979, pp. 318-326.

8. A. Rosenfeld and L.S. Davis, "Iterative histogram modification," IEEE Trans. on SMC, Vol. 8, No. 4, pp. 300-302, 1978.

9. L.S. Davis and A. Rosenfeld, "Cooperating processes for low-level vision: A Survey," Technical Report 123, Jan. 1980, Univ. of Texas at Austin.

2.11 Boundary Detection in Range Pictures*

S. Inokuchi and R. Nevatia

Introduction

Availability of range i.e. the distances of observed points of a scene from the viewer, simplifies many scene analysis problems including segmentation and shape analysis. However, extraction of boundaries of object surfaces is still a non-trivial problem and many techniques have been described in previous work [1-4]. Most of the techniques may be said to use a "region approach"; they attempt to isolate planar regions or grow smooth, curved regions from smaller planar regions. Use of edge like processing has been limited to the use of "jump boundaries", corresponding to a large range discontinuity caused by large separation between objects [1,4].

In this paper, we describe a more comprehensive approach to boundary detection using edge processing. In a previous comparison of segmentation of intensity images, Nevatia and Price conclude that the edge and region techniques are complementary and each is suited to particular tasks [5]. It is hoped that similar observations will hold for range data processing. In particular, the edge methods should work better with thin and long surfaces thus preserving fine detail, and be more adaptable to curved surfaces.

---

*This work is largely supported by a fellowship to Dr. S. Inokuchi, who is visiting USC from Osaka University, Osaka, Japan. It is included here due to possible applications in the DARPA IU work.

Boundary Detection

Our boundary detection process proceeds in the following steps.

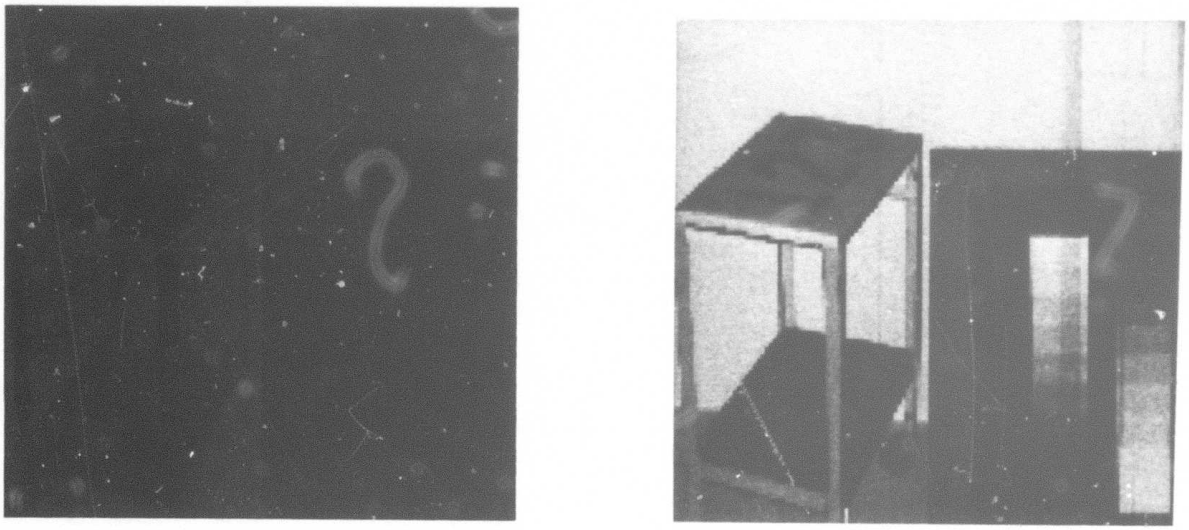
a. Preliminary Edge and Line Detection

The first stage of processing uses an edge and line detection technique developed for intensity images and described in [6]. The image is first convolved with ideal edge masks in six directions, 30 degrees apart, the maximum output determining the edge magnitude and the direction. The edge magnitude and directions are used to obtain thinned and thresholded binary edges. In our experiments, we have used edge masks suitable for detecting step edges. Thus, edges along intersection of two surfaces may not be detected reliably. Also, false edges may be detected on surfaces with a high slope from the viewer. The latter difficulty could be avoided by use of second derivative or "bar" masks. We expect to use such masks for future work. The remainder of the technique is not expected to be affected by this change.

The binary edges are linked to their neighbors with similar edge directions, and finally the linked segments are approximated by piecewise linear segments. Figure 1(a) shows a range image and Fig. 1(b) the corresponding intensity image. Figure 2(a) shows the edges detected in the range image and Fig. 2(b) shows the resulting segments (the numeric labels in Fig. 2(b) are explained later).

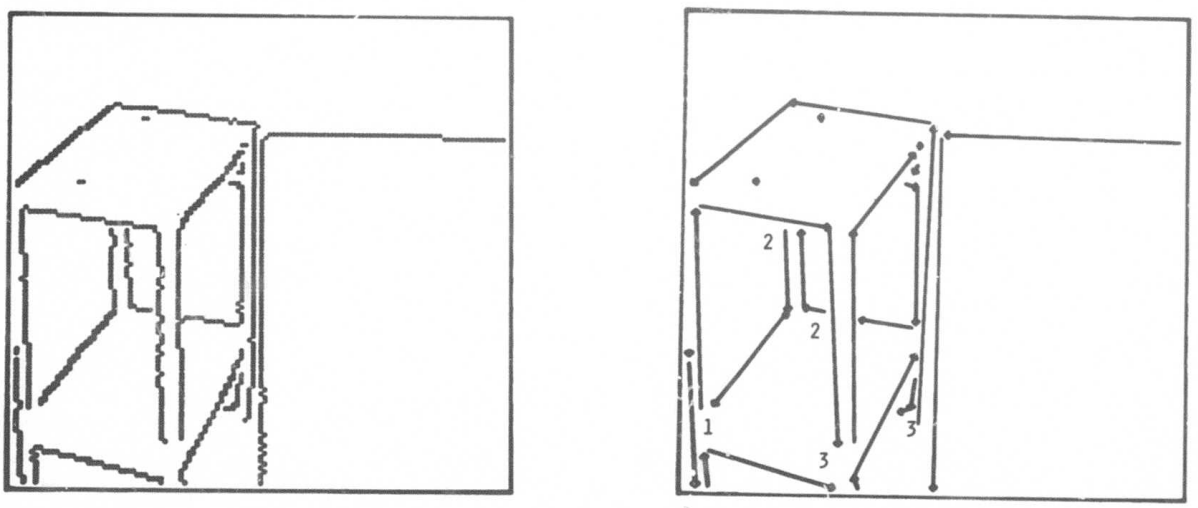
b. Extension of Segments

Fig. 2(b) shows many segments that are incomplete, in that they terminate without connecting to other segments. In usual intensity image processing, proper extension of such segments is a difficult task. In range images, we can use 3-D information to facilitate such extension. The extension proceeds in three steps, corresponding to the three types of "loose ends" labeled in Fig. 2(b).



(a) (b)

Fig. 1. An example (data courtesy of Dr. R.O. Duda, SRI International).



(a) (b)

Fig. 2. Detected edge and lines.

i) Loose ends marked "1" occur near an intersection of two segments. At such points, the local edge directions are unreliable. We extend such segments by ignoring edge direction in the thinning and linking steps in (a) above, and requiring a larger threshold on the edge magnitudes instead.

ii) Loose ends labeled "2" occur near "T" junctions. Again, the connecting of two segments is inhibited in step (a) due to restrictions on local edge directions, which are now relaxed for extension.

iii) Loose ends labeled "3" are more interesting and caused by the 3-D nature of range data. In the examples shown, the detected segments are edges between a surface and another surface far behind it. However, along the extension of the edge segment, the two surfaces come nearer, eventually interesting at a corner, and the desired extensions are not detected as jump boundaries. To extend such segments, we compute 3-D equations of two lines parallel to the segment, but on either sides of it (i.e. lying in two separate planes). If these two lines intersect, and the intersection is along the extrapolated boundary segment, and there exist edge pixels whose 3-D position is near the computed intersection, then the boundary segment is extended.

Figure 3 shows the results of all the above three extensions applied to segments of Fig. 2(b).

c. Radial Line Detection

Some of the interior boundaries, corresponding to intersection of two object surfaces may be still missing. Typically, this happens for short segments at the corners. Now, for each corner, we examine if a line can be found in some direction such that the surfaces on its two

sides correspond to different planes. A window (15x15) is centered around each corner and we find the part of the window containing the object surfaces (by using the orientation of the segments). This region is divided in two subregions R_1 and R_2 by a line through the corner at chosen angles (see Fig. 4(a)). For each choice of θ, planes are fit, in the least mean squared error sense, using the range data. If an additional edge exists at this corner, for some angle, θ, the total error of fitting planes to R_1 and R_2 should be small. If the ratio of the minimum to the maximum error, for different θ, is less than 1/2, we construct a line segment at the angle corresponding to the minimum error. The case of a window containing two corners is shown in Fig. 4(b). The radial line detector is also applied to loose ends (an example is the segment near the wheel of the cart). In this case, the detected line may be anywhere in the window. If the missing segment is long, we need to "follow" the newly detected segment. However, this last step is not necessary for our example and we have not yet implemented such a line following procedure.

Figure 5 shows all segments resulting after application of the radial line detector. Note that the segments detected at this step are undirected (the previous segments are directed in that the object surfaces, assumed to be nearer to the viewer than the background, are to the left of them).

Our boundary detection technique is similar in spirit to the heterarchical technique of Shirai for intensity images of polyhedral objects [7].

Region Descriptions

Finding boundaries of closed regions is relatively simple, if complete boundary segments are found. We are currently finishing the region tracing programs. Interestingly, some region properties can be inferred by observing the orientation of segments along a region as shown in Fig. 6. Basically, the parts and spaces can be

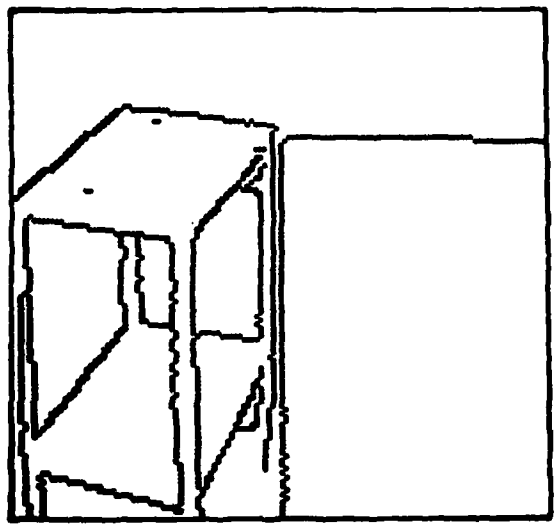


Fig. 3. Edges after extension of line segments.

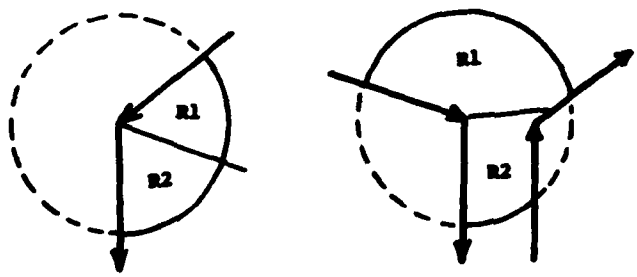


Fig. 4. Radial line detection a) and b).

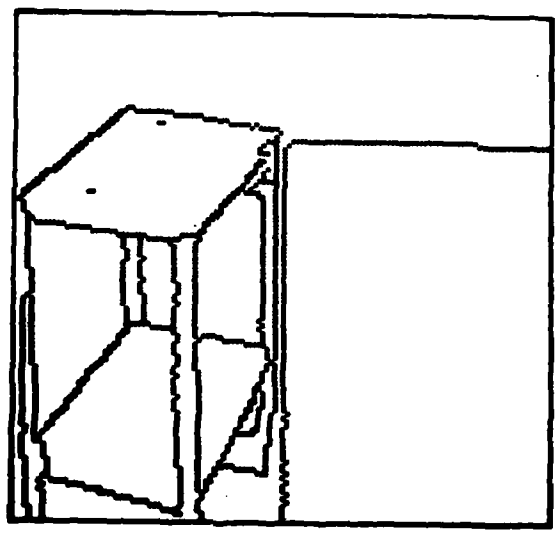


Fig. 5. Edges after radial line detection.

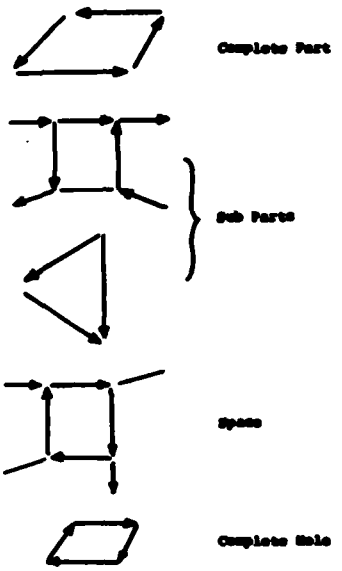


Fig. 6. Types of regions.

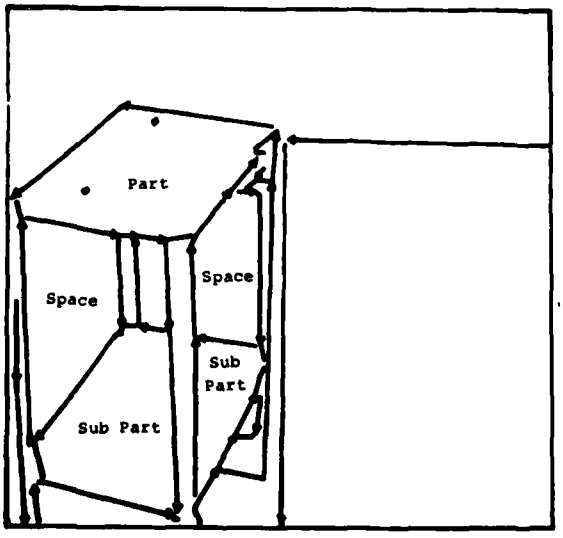


Fig. 7. Regions in the earlier example.

differentiated by whether the region is traced in clockwise or countr-clockwise direction. Occlusion is indicated by not all segments pointing in the same direction, compared to the direction of region tracing. Related line labeling analysis may be found in [8]. Figure 7 shows some of the regions in our example that can be inferred from them. This part of our process is currently being implemented. Eventually, we hope to better describe occluded sub-parts, and be able to merge such sub-parts when appropriate.

Conclusions

We have described a technique to extract fairly complete boundaries from range data, exploiting the properties of such data. Detailed comparative evaluations with region based techniques are yet to be made. However, we expect our technique to be complementary and more suitable for certain tasks.

References

1. D. Nitzan, A.E. Brain and R.O Duda, "The Measurement and Use of Registered Reflectance and Range Data in Scene Analysis," Proc. IEEE, Vol. 69, No. 2, Feb. 77, pp. 206-220.

2. R.O. Duda, D. Nitzan and P. Barrett, "Use of Range and Reflectance Data to Find Planar Surface Regions," IEEE Trans. PAMI, Vol. 1, No. 3, July 1979, pp. 259-271.

3. M. Oshima and Y. Shirai, "A Scene Description Method Using Three-Dimensional Information," Pattern Recognition Journal, 1979, pp. 9-17.

4. R. Nevatia and T.O. Binford, "Description and Recognition of Curved Objects," Artificial Intelligence, Vol. 8, No. 1, Feb. 77, pp. 77-98.

5. R. Nevatia and K. Price, "Locating Structures in Aerial Images," Proc. of the IJCPR, Kyoto, Japan, Nov. 78 pp. 686-690.

6. R. Nevatia and K.R. Babu, "Linear Feature Extraction and Description," Proc. of IJCAI-79, Tokyo, Japan, August 1979, pp. 639-641.

7. Y. Shirai, "Analyzing Intensity Arrays Using Knowledge About Scenes," in The Psychology of Computer Vision, P.H. Winston, Editor, McGraw-Hill, 1975.

8. K. Sugihara, "Range Data Analysis Guided by a Junction Dictionary," Artificial Intelligence Journal, Vol. 12, 1979, pp. 41-69.

3. HARDWARE IMPLEMENTATION OF IU ALGORITHMS

3.1 ADVANCED IMAGE UNDERSTANDING USING LSI AND VLSI

S.D. Fouse, G.R. Nudd, and V.S. Wong

Hughes Research Laboratories
Malibu, California 90265

Abstract

We describe here the work undertaken at the Hughes Research Laboratories, Malibu, in support of the DARPA Image Understanding (IU) program. This report covers the period from October 1979 through April 1980, and as such represents a transition period for us. Our work prior to this period has been concerned with the investigation and demonstration of large scale integration (LSI) microelectronic technology for image understanding. The principal aims of this work were the design, fabrication of and demonstration of high-speed primitives for real-time processing with low-level operators. This work is continuing in that we are completing a performance evaluation of the fourteen low-level operators already implemented on the program, and interacting with other groups in the program, (including USC, MIT, and Stanford) is determined how these might perform in a full-scale IU system. The second issue which is now becoming the mainstream of the program is a detached investigation of the applicability of very large scale integration (VLSI, >50,000 gates/chip) to higher level systems. The aims and our progress on this is described below.

I. INTRODUCTION

Our work this period on the image understanding (IU) program has been divided into two distinct topics: the performance investigation of the LSI circuit developed for the low-lead operators and an investigation and analysis of the potential of VLSI for IU. We are starting a detailed performance review to characterize the throughput accuracy and dynamic range of each of the 14 primitives developed to date on the program. This will enable us to determine how they might interface with either a host machine or operate as part of a more complex VLSI system. (To this end, we are supporting both USC and the Artificial Intelligence Laboratory of MIT in their vision projects.)

The second topic dealing with the application of VLSI to IU has involved three issues, on each of which we have made some progress: (1) a directed graph analysis of candidate systems, (2) an investigation of the applicability of residue operations for VLSI, (3) development of a simulation capability for the processors. Initially, we have chosen to look at three systems: a line finder, texture analyzer, and a segmentation scheme. Details of each topic are given below.

II. TEST CHIP III PERFORMANCE EVALUATION

One aspect of our work this year has been to continue the testing and evaluation of the third test chip, a CCD-MOS design. This chip, as previously described in Ref. 1, has five functions: a 5x5 programmable convolution, a 7x7 mask programmable convolution currently implemented as an edge detector, a 3x3 Laplacian operator, a 5-element sort for median filtering, and a 26x26 bipolar convolution. Preliminary test results[2] demonstrated that the circuits were functionally correct (e.g., the 5x5 kernel was programmable, the sort circuit could perform the sort, and the 26x26 convolution had the proper impulse response). Results included images that were processed by the chip at pixel rates of $\sim$20 kHz. Our current work has been to extend the testing, both in terms of quantitative performance evaluation and in terms of achievable pixel rate.

Some of the plans we have for this comprehensive testing include:

(1) Calculation of dynamic range of programmable weights

(2) Determination of linearity of weights

(3) Experimental analysis of sample-and-hold circuit

(4) Dynamic range and linearity of sort circuit

(5) Performance at higher speeds.

To help complete this evaluation in a timely manner, we have trained a technician to operate these circuits. This should enable a much more thorough job to be done. In addition, a new lot of chips is currently being processed, giving us a larger sample of the circuits to work with as well as possibly improved chips.

III. DEVELOPMENT OF INTERMEDIATE-LEVEL VLSI IMAGE-UNDERSTANDING SYSTEM

A major goal of the current phase of our work on hardware for IU systems is to determine the impact that VLSI technology can have in this area. Our method for accomplishing this is shown in Figure 1. We have selected candidate IU systems that extend beyond the low-level processing which has been implemented previously. The selected systems have been characterized using directed graphs, which help to make explicit the parallelism that is inherent in the algorithms. The graphs will also aid in determining the commonality between the system, thus enabling us to select a partioning in which functions constructed can be used, as is, in several systems.

In addition to our directed-graph analysis, we have also been working in two areas to support the development of VLSI architectures: residue computation techniques and the acquisition and familiarization of a computer simulation program (ECSS). The work on residue computation as well as future work in defining an operating environment (e.g., will this processor be used as a peripheral to a general-purpose computer or as a stand-alone system), will be fed into the selection of candidate architectures. Following the selection of the initial architectures, an

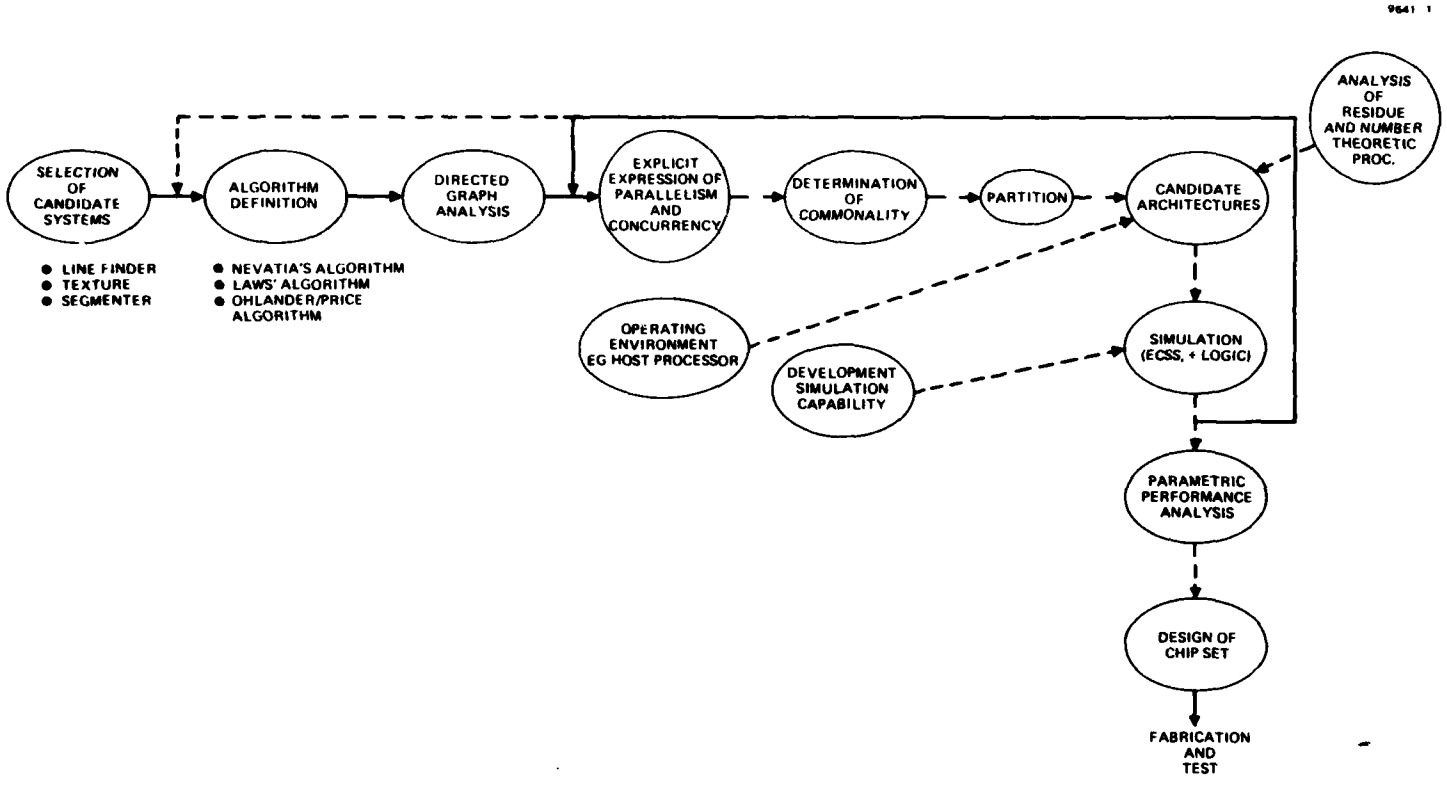


Figure 1. Program plan for development of intermediate-level VLSI image understanding system.

iterative process of simulation using ECSS, performance analysis, and re-specification of the architecture will be undertaken. Once this process converges to an optimal design, we will take the further step of generating a fairly detailed design of a chip set for one of the algorithms.

A more detailed description of the work that has been done to date on this plan follows, including as well a preliminary sizing of one of the candidate systems. This was included to provide a little more insight into the scale of the problem we are trying to tackle.

A. Directed Graph Analysis of Candidate Systems

Three systems involving intermediate load processing were selected for the purpose of developing architectures and utilizing commonality between them. The systems chosen were a line-finding system, a texture-classification system, and a segmentation system. To be able to perform the directed graph analysis, algorithms must be specified; the following algorithms were chosen:

(1) Nevatia line finder

(2) Laws texture classification system

(3) Ohlander/Price segmenter.

The graph analysis was carried out using papers describing the algorithms and by talking to the individuals involved in their development.[3,4,5] The graphs for the three systems are shown in Figures 2, 3, and 4. Constructing the graphs involves identifying logical function blocks and the data inputs and outputs of each block. To indicate the throughput required, the word length and the data rates are indicated on the arcs connecting the function blocks. For example, for the Nevatia line finder in Figure 3, the input data is an eight-bit parallel word and the data rate is NxN pixels per frame multiplied by F frames per second.

Once the graphs are constructed, we can begin to see the parallelism potentials of the algorithms as well as the common functions between the systems. For example, for the Laws texture system, it is apparent that the processing from the 5x5 convolutions through to the energy measure

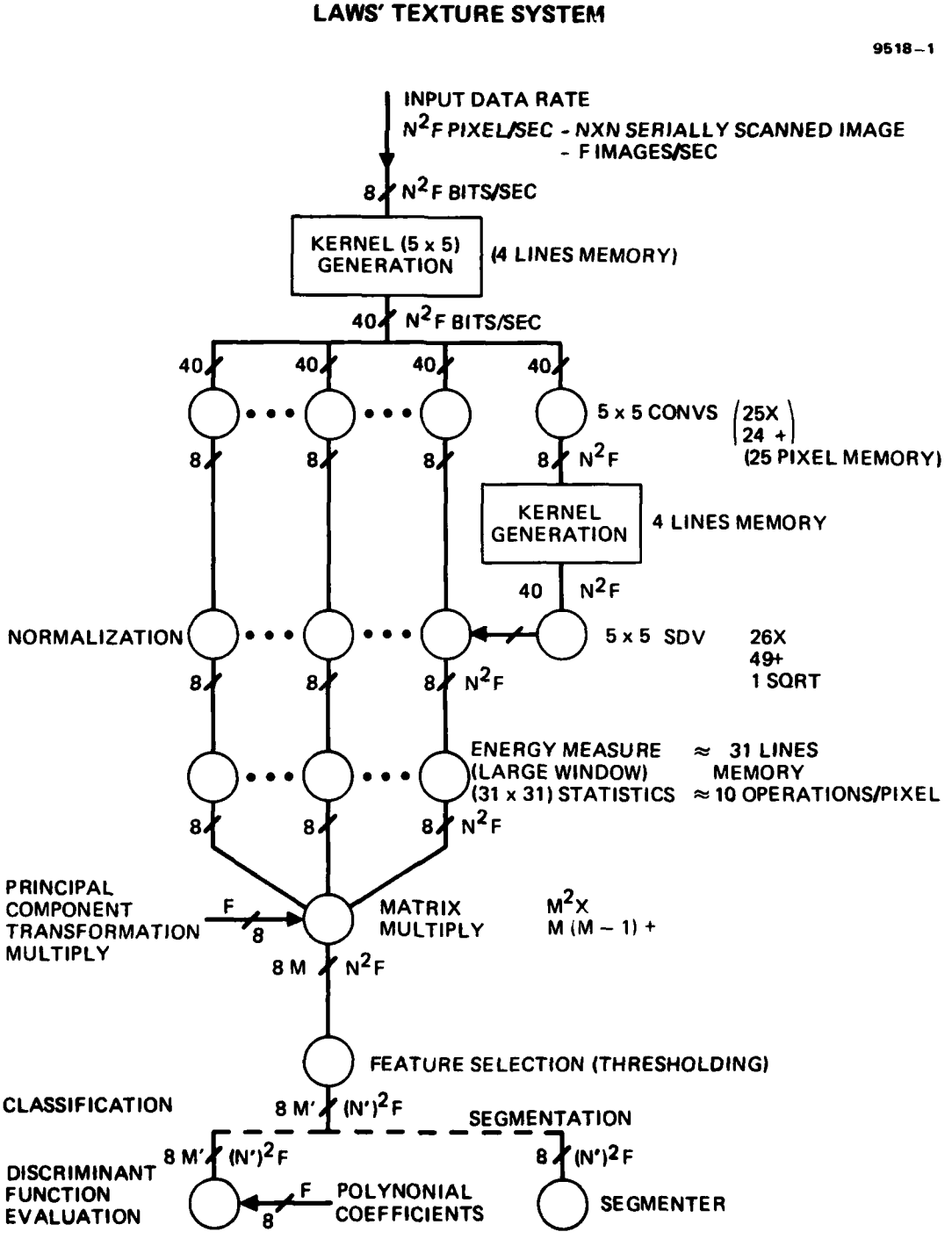


Figure 2. Laws texture classification system.

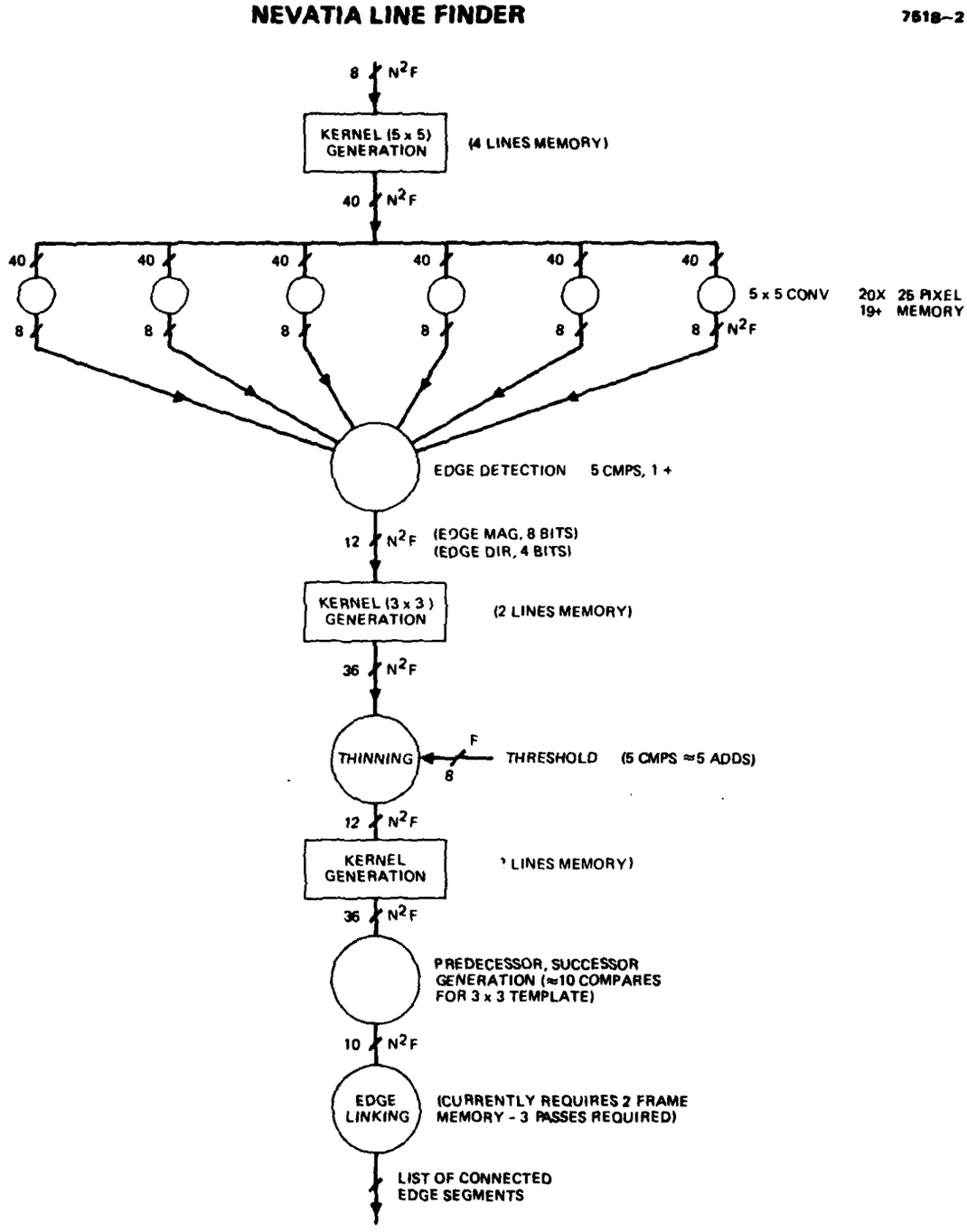


Figure 3. Nevatia line finder.

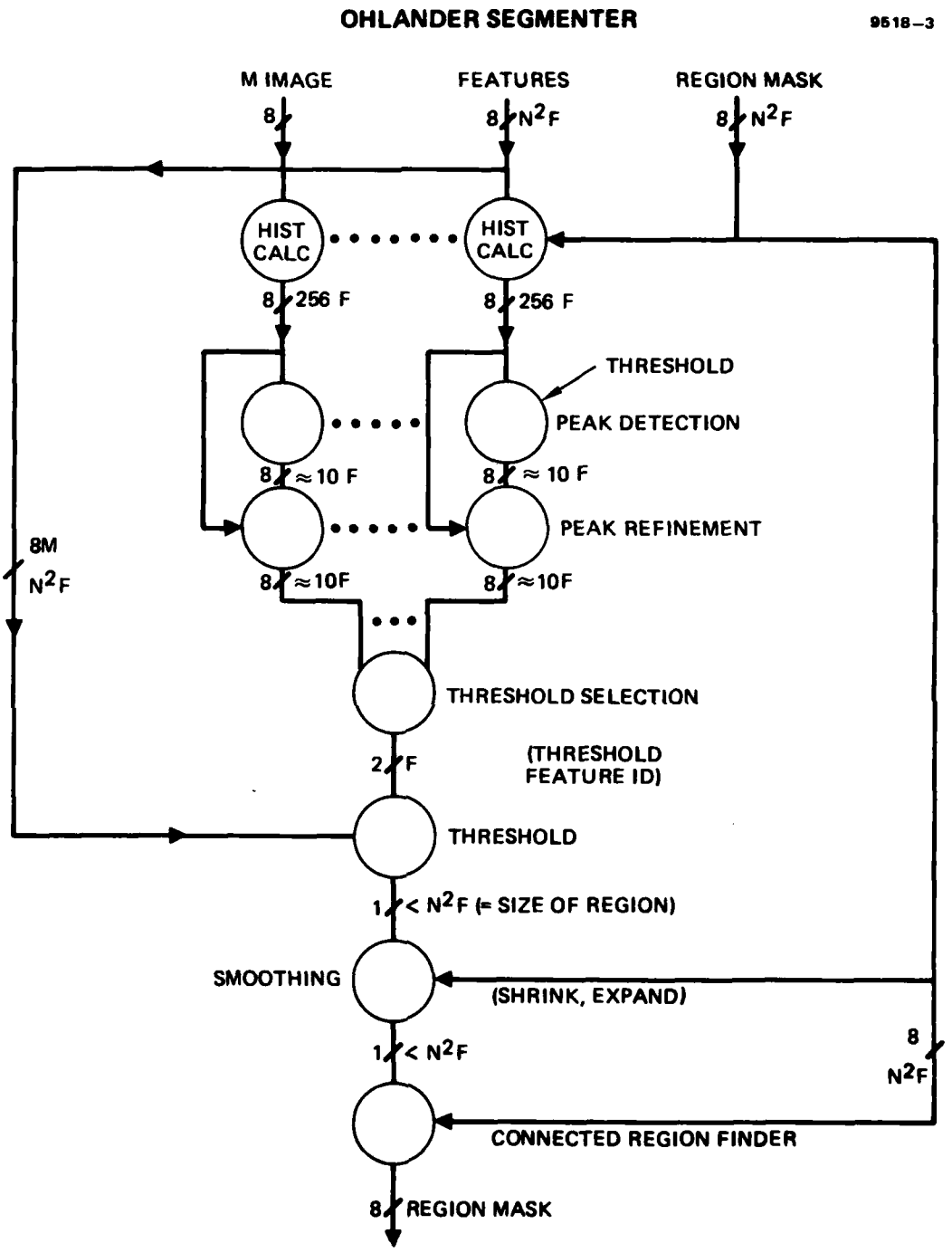


Figure 4. Ohlander/Price segmenter.

involves M independent signal paths, indicating that a logicalpartition would be to put the 5x5 convolution, the normalization, and the large window energy measure on a single chip (if possible); to realize this system, we need only make M copies of the chip. The advantage of this type of partitioning is that the communication to and from the chip is minimized. The several similarities between the systems are made obvious by the graphs. There are parallel convolutions in both the line-finder and the texture systems. A less obvious similarity is between the edge-linker function in the line finder and the connected region finder in the segmentation system. Both functions play the same role of linking data points into a single element, either a line or a region.

B. Residue Techniques for an IU Processor

To appreciate our interest in residue arithmetic and its application to IU, it is necessary to understand the mathematical foundations. To explain it, some mathematical notation must be introduced. The expression "A Mod M" is equal to the remainder when A is divided by M. Both A and M must be integers and A mod M must also be an integer between 0 and M-1. Using modern algebra, it is possible to show two facts. First is the expression

$$((A \bmod M) + (B \bmod M)) \bmod M = (A + B) \bmod M .$$

This implies that we can calculate the remainder of a sum by adding the remainders themselves. The second fact is that the k-tuple (A mod M1, A mod M2,...,A mod Mk) is a unique representation of A if two conditions are met: the M_i's are relatively prime, and A is less than the product of the M_i's. These two conditions allow two large numbers to be added by adding several pairs of smaller numbers, the smaller numbers being remainders. This can also be said for multiplication.

To perform residue operations, three steps are involved. First, the binary operands must be converted to remainders of several bases (M_i's). Next, these remainders must be added in a modular fashion (i.e., the sum of the two remainders must also be a remainder, which is

to say that one must take the remainder of the sum). Finally, the result must be converted back to binary, which seems to be the most difficult of the three steps. In addition to this overhead of conversion, it is difficult to make decisions while in the residue representation since, unlike binary, there is no ordering associated with the representation.

The overhead costs associated with the residue technique for computation makes it inappropriate for many applications. But often it is easier to do several operations with small operands than to do one operation with large operands. So if enough operations can be done between the two conversion steps, the process as a whole could be cheaper to do than the equivalent process using binary arithmetic. For IU work, the bulk of the computation is done at the low level and consists mostly of signal processing requiring no decisions to be made. For algorithms such as a line finder that require several parallel convolutions, there are numerous operations that can be performed in the residue form. Additionally, we are dealing with integer quantities, which are ideally suited to residue techniques. In terms of building hardware, a residue processor would have relatively simple logic performing the bulk of the operations and the complex logic would be concentrated at the input and the output of the system. This is an advantage in terms of reliability and testability.

Our goal for this program is to examine the different systems that we have chosen and to determine the feasibility of using the residue technique. The study will be done in a parametric fashion, so that, by using the parameters of a particular system, one can determine if residue will be an advantage. Areas that we have already begun to look at or will be looking at include conversion logic, dynamic range requirements (rounding in residue makes little sense), logic for adding and multiplying in residue, methods for scaling intermediate results, and methods of representing the numbers in residue to facilitate the logic for calculations.

C. ECSS Simulation

As the density of micro-electronic circuits increases, the systems that can be built on these chips may become very sophisticated. To

optimize the design and to be able to predict performance, it is desirable to simulate the operation before the circuits are actually constructed in hardware. For our work on systems design at HRL, we have chosen to use a program developed specifically for computer system simulation at the RAND Corporation called ECSS (Extendable Computer Simulation System).

ECSS is written in a natural, English-like language, and it is both descriptive and flexible to use. It stresses a procedural format, and the syntax of the language is constructed so that it can compactly define and specify the components of a computer system (storage devices, I/O devices, CPUs, printers) as well as interactions among the components (job scheduling, resource allocation, interprocess communication). To indicate performance and bottlenecks of the computer system being modeled, ECSS outputs statistics showing the jobs activated, percent utilization of the devices, and length of queues for the various resources.

The main advantages of using ECSS over other simulation programs include:

- Flexibility. ECSS provides a variety of techniques for modeling systems. The user, depending on his interests, may focus on modeling program behavior, system I/O transfers, or arbitrary activities running on a particular device.

- Extendability. ECSS is built on SIMSCRIPT II, a program language designed for discrete event simulation. Therefore, ECSS has all the provisions of data types and procedural statements of SIMSCRIPT. This allows the user to extend definitions of ECSS structures (devices, jobs, transmissions) and add new commands as desired.

- Programmability. When the models provided by ECSS do not meet the requirements of the user, it is possible to modify the source code for ECSS. The service routines are written in modular fashion in SIMSCRIPT, thus facilitating modification.

At HRL, we are presently using ECSS to simulate computer architectures for our IU program and to investigate specialized VLSI designs for a line finder, a texture system, and a segmentation system. We anticipate that ECSS will be a very useful tool for developing the architectures, predicting the performance of the systems, and in general for communicating and documenting the designs. We also have interest in the

area of specialized cellular array processors for executing concurrent algorithms. In this respect, we believe that ECSS would be very useful in the design of such an array processor.

D. Preliminary Chip Sizing of Line-Finder System

To obtain a clearer idea of the magnitude of the problem we are trying to solve, as well as to determine the pacing items for fabricating the system, we have decided to determine roughly how many gates the line-finder system would need. The gate count was divided into gates for memory and gates for random logic. We then used these figures, along with a state-of-the-art number for gates per chip for both random logic and memory, to determine an approximate total number of chips for the system. Additionally, we calculated a chip count for the predicted 1985-1990 technology.

The gate count for the line finder was done for each block on the directed graph. These functions include:

(1) Kernel generation

(2) 5x5 convolutions

(3) Edge detection

(4) Edge thinning

(5) Predesessor and successor generation

(6) P and S memory

(7) Edge linking.

Rough designs were performed for the convolutions, edge detection, and edge thinning to estimate the number of gates. The P and S generation can be estimated by replicating the edge-thinning logic six to ten times, since it is making similar comparisons as for edge thinning but more of them. The logic for edge linking gets very involved, since it must be able to generate addresses for random access of the P and S memory. In addition, logic for each pass must be generated, since each pass performs a different function. So for the purposes of this sizing, the edge

linking was not estimated. Table 1 shows the gate counts for each of the functions. Notice that, for the convolutions, we sized two types, one with no multiplies and one with six multiplies. This is because the algorithm we are sizing has six convolutions, two which can be scaled such that all the weights are either zero or one, and four which can be scaled such that all but six of the weights are either zero or one.

The current state of the art for MOS technology allows us to put 64K bits of memory or 20,000 gates of random logic on a single chip. Using these figures, we are able to partition the algorithm onto a set of chips and thus calculate a total number of chips for the system. One possible partitioning is as follows:

CHIP	FUNCTIONS
1	5 line kernel, 2 convolutions (no multiply)
2-5	One 5x5 convolution (6 multiplies)
6	Edge detection, 3 line kernel, thinning logic
7	3 line kernel, P and S generation
8-27	P and S memory (40 64K chips)
48	Edge linking logic.

It is fairly obvious to see that the two pacing items (not counting the edge-linking logic, which we have not looked at) are the P and S memory and the convolution calculations. The convolution complexity can be simplified by reducing the accuracy required on the weights of the kernel. This would mean that a logic multiplier could be fast enough, and the memory for look up table multiplies would not be needed. The requirement for the P and S memory is embedded in the algorithm, and thus the algorithm would need to be altered if we wanted to reduce this memory.

We can perform this same analysis assuming some future technology. We expect that in several years we will be able to achieve 1 Mbit of memory or 0.5 million gates of random logic on a single chip. This assumes 1-μm feature size as compared to 5-μm features for current technology. For this case, we get the following partitioning:

CHIP	FUNCTION
1	All random logic and kernel generation memory
2-4	P and S memory.

Here we can have 8 lines for the input, 8 lines for the output, and some address and data lines for the P and S memory, avoiding any pin-out problems.

The papers presented as Appendices A and B were prepared during this quarter.

Table 1. Gate Count for Line Finder

Function	Gate Count	Number Used	Total
5 line kernel	4x512[1]x8 = 16K bits	1	16K bits memory
5x5 convolution no multiplies	1.25K	2	2.5K
5x5 convolution 6 multiplies[2]	13.25K	4	53K
Edge detection	0.35K	1	0.35K
3 line kernel	2x512x12 = 6K bits	2	12K bits memory
Edge thinning	0.35K	1	0.35K
P and S generation	3K	1	3K
P and S memory	512x512x10 = 2.5 Mbit	1	2.5 Mbit memory

1 - Assumed image size = 512x512
2 - Multiplies accomplished using 8x256 bit ROM.

The following papers describing this work were prepared during the last six months.

S.D. House, G.R. Nudd and P.A. Nygaard, "Implementation of Image Preprocessing Functions Using CCD LSI Circuits," Proc. SPIE Tech. Symp. - IR Image Sensor Technology, Vol. 225, Washington, D.C., April 1980.

G. Nudd, "Image Understanding Architectures," National Computer Conference, Anaheim, Ca., May 1980.

REFERENCES

1. G.R. Nudd, P.A. Nygaard, S.D. Fouse, and T.A. Nussmeier, "Implementation of Advanced Real-Time Image Understanding Algorithms," Proceedings Image Understanding Workshop, DARPA, April 1979.

2. G.R. Nudd, S.D. Fouse, T.A. Nussmeier, and P.A. Nygaard, "Development of Custom-Designed Integrated Circuits for Image Understanding," Proceedings of Image Understanding Workshop, DARPA, November 1979.

3. R. Nevatia and K. Ramesh Babu, "An Edge Detection, Linking, and Line Finding Routine," USC IPI Semiannual Report, September 1978.

4. K.I. Laws, "Textured Image Segmentation," Ph.D. Thesis, USC, January 1980.

5. R. Ohlander, K. Price, and D. Raj Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," Computer Graphics and Image Processing, 1978.

4. RECENT INSTITUTE PERSONNEL PUBLICATIONS AND PRESENTATIONS

1. A. Armand, D. Boswell, A.A. Sawchuk, B.H. Soffer, T.C. Strand, "Real-Time Parallel Optical Analog-to-Digital Conversion," Optics Letters, Vol. 5, March 1980.

2. A. Armand, A.A. Sawchuk, T.C. Strand, "Nonlinear Optical Processing with Halftones: Accurate Predictions for Degradation and Compensation," submitted to Applied Optics.

3. A. Armand, A.A. Sawchuk, T.C. Strand, "Real-Time Parallel Logarithmic Filtering," submitted to Applied Optics.

4. J. Bescos, I. Glaser, A.A. Sawchuk, "Restoration of Color Images Degraded by Chromatic Aberrations," submitted to Applied Optics, Vol. 19.

5. D. Boswell, P. Chavel, A.M. Lackner, A.A. Sawchuk, B.H. Soffer, T.C. Strand, A.R. Tanguay, Jr., "Optical Computing with Variable-Grating Mode Liquid Crystal Light Valves," Proceedings 1980 International Optical Computing Conference/SPIE Proceedings, Vol. 232, Washington, D.C., April 1980.

6. D. Boswell, P. Chavel, A.A. Sawchuk, B.H. Soffer, T.C. Strand, A.R. Tanguay, Jr., "Parallel Optical Analog-to-Digital Conversion Using a Liquid Crystal Light Valve," Workshop on High Speed A/D Conversion, Portland, Oregon, February 1980.

7. D. Boswell, A.M. Lackner, A.A. Sawchuk, B.H. Soffer, T.C. Strand, A.R. Tanguay, Jr.. "Variable Grating Mode Liquid Crystal Device for Optical Processing and Computing," Eighth International Liquid Crystal Conference, Kyoto, Japan, June-July 1980.

8. D. Boswell, A.A. Sawchuk, B.H. Soffer, T.C. Strand, A.R. Tanguay, Jr., "Variable Grating Mode Liquid Crystal Device for optical Processing," Proceedings Society of Photo-Optical Instrumentation Engineers Los Angeles Technical Symposium - Devices and Systems for Optical Signal Processing, Vol. 218, Los Angeles, February 1980.

9. O.D. Faugeras, "An Overview of Probabilistic Relaxation Theory and Applications," Proceedings of the NATO ASI, D. Reidel Publishing Company, June 1980.

10. O.D. Faugeras, "Autoregressive Modeling with Conditional Expectations for Texture Synthesis," submitted to the 5th ICPR, March 1980.

11. O.D. Faugeras, "Decomposition and Decentralization Techniques in Relaxation Labeling," submitted to Computer Graphics and Image Processing, March 1980.

12. O.D. Faugeras, "Improving Consistency and Reducing Ambiguity in Stochastic Labeling: An Optimization Approach," to be published in the IEEE PAMI Transactions, 1980 (with M. Berthod).

13. O.D. Faugeras, "Scene Labeling: An Optimization Approach," to be published in Pattern Recognition, 1980 (with M. Berthod).

14. O.D. Faugeras, "Using Context in the Global Recognition of a Set of Objects: An Optimization Approach," to be published in the Proceedings of the 8th World Computer Congress (IFIP 80), (with M. Berthod).

15. O.D. Faugeras, D.D. Garber, "Algebraic Reconstruction Techniques for Texture Synthesis," submitted to the 5th ICPR, March 1980.

16. O.D. Faugeras, W.K. Pratt, "Decorrelation Methods of Texture Feature Extraction," to be published in the IEEE PAMI Transactions, 1980.

17. O.D. Faugeras, K.E. Price, "Semantic Description of Aerial Images with Stochastic Labeling," submitted to the 5th ICPR, March 1980.

18. L.M. Frantz, A.A. Sawchuk, W. Von der Ohe, "Optical Phase Measurement in Real Time," Applied Optics, Vol. 18, pp. 3301-3306, October 1979.

19. S. Inokuchi, and R. Nevatia, "Boundary Detection in Range Data," submitted for conference presentation.

20. C.M. Lo and A.A. Sawchuk, "Nonlinear Restoration of Blurred Images with Poisson Noise," 1979 Annual Meeting, Optical Society of America, Vol. 69, pp. 1441-1442, Rochester, New York, October 1979.

21. C.M. Lo, A.A. Sawchuk, "Restoration with Poisson Noise," Proceedings of the IEEE, Vol. 68, 1980 (invited paper to appear).

22. J. Mantock, A.A. Sawchuk, T.C. Strand, "An Optical Processor Applied to Cloud Classification," Proceedings Society of Photo-Optical Instrumentation Engineers Los Angeles Technical Symposium - Devices and Systems for Optical Signal Processing, Vol. 218, Los Angeles, February 1980.

23. J. Mantock, A.A. Sawchuk, T.C. Strand, "Hybrid Optical-Digital Texture Analysis," First ASSP Workshop on Two-Dimensional Digital Signal Processing, Berkeley, California, October 1979.

24. J. Mantock, A.A. Sawchuk, T.C. Strand, "Hybrid Optical/Digital Texture Analysis," Optical Engineering, Vol. 19, March/April 1980, (invited paper).

25. J. Michaelson, A.A. Sawchuk, "Nonlinear Optical Processing Using Liquid Crystal Light Valves," Proceedings Society of Photo-Optical Instrumentation Engineers Los Angeles Technical Symposium - Devices and Systems for Optical Signal Processing, Vol. 218, Los Angeles, February 1980.

26. R. Nevatia, "Image Understanding Research at USC," Seminar presented at University of California, Irvin, Ca., March 1980.

27. R. Nevatia, "Matching of Natural Terrain Scenes," with C. Clark, D. Conti, W. Eckhardt, T. McCullogh and D. Tseng, submitted for conference presentation.

28. R. Nevatia and K.R. Babu, "Linear Feature Extraction and Description," accepted for publication in Computer Graphics and Image Processing.

29. R. Nevatia, and K.E. Price, "Locating Structures in Aerial Images," submitted for Journal publication.

30. R. Nevatia and K.E. Price, "Symbolic Representation in USC IU System," Proc. of ARPA Image Understanding Workshop, Los Angeles, Ca., November 1979.

31. R. Nevatia and A.A. Sawchuk, "Image Processing and Understanding Research," Seminar at Battelle Memorial Institute, Richland, Wash., March 1980.

32. R. Nevatia and A.A. Sawchuk, "Progress in Image Understanding Research at USC," Proceedings Image Understanding Workshop, pp. 149-151, Los Angeles, Ca., November 1979.

33. F. Vilnrotter, R. Nevatia and K.E. Price, "Structural Description of Natural Textures," submitted for conference presentation.

5. RECENT PH.D. DISSERTATIONS

The abstracts of recent Ph.D. Dissertations completed with DARPA support are listed in this section. The dissertations are published as USCIPI technical reports.

5.1 Textured Image Segmentation

Kenneth Ivan Laws

USCIPI Report 940, January 1980

The problem of image texture analysis is introduced, and existing approaches are surveyed. An empirical evaluation method is applied to two texture measurement systems, co-occurrence statistics and augmented correlation statistics. An "spatial-statistical" class of texture measures is then defined and evaluated. It leads to a simple class of "texture energy" transforms, which perform better than any of the preceding methods. These transforms are very fast, and can be made invariant to changes in luminance, contrast, and rotation without histogram equalization or other preprocessing.

Texture energy is measured by filtering with small masks, typically 5x5, then with a moving-window average of the absolute image values. This method, similar to human visual processing, is appropriate for textures with short coherent length or correlation distance. The filter masks are integer-valued and separable, and can be implemented with one-dimensional or 3x3 convolutions. The averaging operation is also very fast, with computing time independent of window size.

Texture energy planes may be linearly combined to form a smaller number of discriminant planes. These principal component planes seem to represent natural texture dimensions, and to be more reliable texture measures than the texture energy planes.

Texture segmentation or classification may be accomplished using either texture energy or principal component planes as input. This study classified 15x15 blocks of eight natural textures. Accuracies of 72% were achieved with co-occurrence statistics, 65% with augmented correlation statistics, and 94% with texture energy statistics.

5.2 Design of SVD/SGK Convolution Filters for Image Processing

Sang Uk Lee

USCIPI Report 950, January 1980

This dissertation describes a special-purpose signal processor for performing two-dimensional convolution with a minimum amount of hardware using the concepts of singular value decomposition (SVD) and small generating kernel (SGK) convolution. The SVD of an impulse response of a two-dimensional finite impulse response (FIR) filter is employed to decompose a filter into a sum of two-dimensional separable linear operators. These linear operators are themselves decomposed into a sequence of small kernel convolution operators. The SVD expansion can be truncated to a relatively few terms without significantly affecting the filter output.

A statistical analysis of finite word-length effects in SVD/SGK convolution is presented. Two important issues, related to the

implementation of the filters in cascade form, scaling and section ordering, are also considered.

Computer simulation of image convolution indicates that 12 bits are required for the SVD/SGK accumulator memory and 16 bits are required for quantization of filter coefficients to obtain results visually indistinguishable from full precision computation. A normalized mean square error between the SVD/SGK processed output and the direct processed output is chosen as an objective criterion function. It is shown that a subjective visual improvement is obtained by resetting the output mean to be equal to the input mean.

The transformation technique developed for the one-dimensional case is used to parametrically modify the cutoff frequency of a baseline SVD/SGK convolution filter. A detailed discussion of the one-dimensional case is presented, and its applicability to SVD/SGK convolution filters is described.